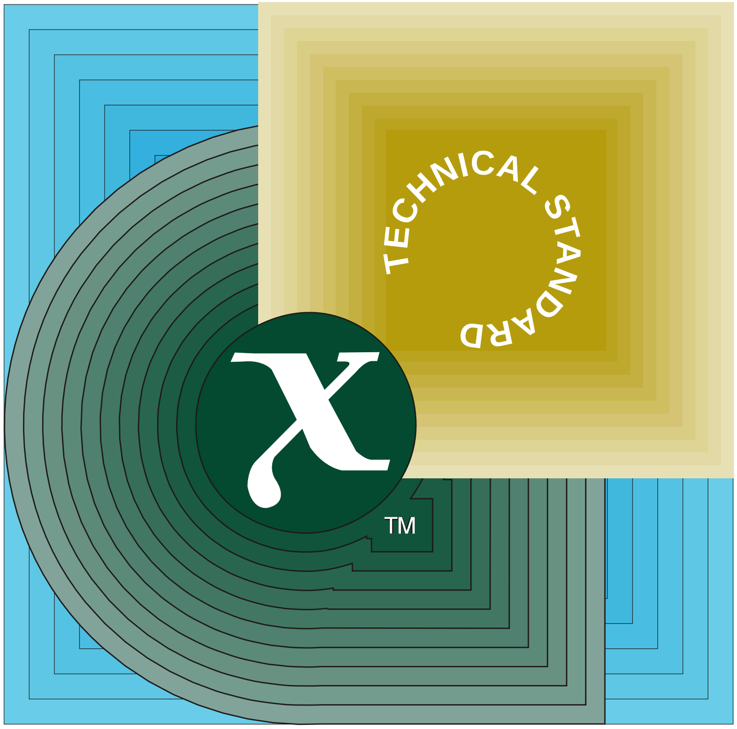
Technical Standard

Byte Stream File Transfer (BSFT)



THE Open GROUP

[This page intentionally left blank]

X/Open CAE Specification

Byte Stream File Transfer (BSFT)

X/Open Company, Ltd.

© *November 1991, X/Open Company Limited*

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of the copyright owners.

X/Open CAE Specification

Byte Stream File Transfer

ISBN: 1 872630 27 8
X/Open Document Number: XO/CAE/91/400

Set in Palatino by X/Open Company Ltd., U.K.

Published by X/Open Company Ltd., U.K.

Any comments relating to the material contained in this document may be submitted to X/Open at:

X/Open Company Limited
Apex Plaza
Forbury Road
Reading
Berkshire, RG1 1AX
United Kingdom

or by Electronic Mail to:

XoSpecs@xopen.co.uk

Contents

BYTE STREAM FILE TRANSFER

Preface

This Document

This document is an XPG CAE Specification (see above). It is a specification of Byte Stream File Transfer (BSFT), an X/Open networking service that provides the means of transferring unstructured (byte-stream) files between X/Open-compliant systems.

BSFT is part of X/Open's strategy for coexistence and migration of users between the Internet Protocol Suite (often referred to as ''TCP/IP'') and OSI. The BSFT interface has been derived from that used by the File Transfer Protocol (FTP) of IPS, but the protocol profile used is from OSI. A user familiar with FTP will thus be able to find almost identical functionality and interfacing in an OSI network running BSFT.

BSFT is fully conformant with the ISO standard for file transfer - **IS 8571**, **File Transfer, Access and Management (FTAM)**.

The FTAM standard has a rich set of functions and options, and, in order to encourage interoperability, regional groups (EWOS, NIST, AOW) and ISO have defined standard subsets of FTAM, known as ''profiles''. This version of the BSFT specification is based on International Standard Profile ISP 10607:1990, File Transfer Access and Management, part 3 (AFT11 - Simple File Transfer Service) and part 6 (AFT3 - File Management Service).

BSFT is thus based on international standards that have reached full IS status, and on internationally harmonised profiles.

Although BSFT is primarily intended to support the transfer and management of unstructured files between X/Open-compliant systems, the use of international standards and profiles allows file transfer and management between a BSFT system and any other system (whether it is X/Open-compliant or not) that supports the base profiles.

BSFT does not specify how FTAM will be implemented: it simply defines the functionality that is visible externally to the machine (i.e., when a machine is communicating with another machine using FTAM) and the functionality that is presented to the user.

A key objective of BSFT is to simplify the simultaneous use of file transfer facilities in the IPS and OSI environments, and to facilitate the coexistence and migration between these two protocol sets by defining a similar user interface and functionality in the two environments.

This version of the BSFT specification provides the functionality of the Internet *FTP* utility, using ISO protocol stacks. In selecting FTAM functionality to be included in BSFT, the general principle has been to include all mandatory features of ISP 10607-3 and ISP 10607-6 and to include optional features only if they are required to produce *FTP* equivalent functionality.

This document does not reproduce the base profiles, instead it indicates which optional features must be supported and, where appropriate, how the required FTAM functions can be mapped onto X/Open operating system functions. Therefore, implementors must have access to the referenced ISPs.

The document comprises introductory, overview and definitive sections.

The definitive sections are:

- **Appendix A**, **BSFT User Interface Definition**
- **Section 7.6**, **File Attributes** through **Section 7.9**, **Invalid Filenames** (mapping to the Responder's filestore)
- **Section 7.9**, **Invalid Filenames** through **Section 7.11**, **Reading files from a Responder** (Initiator's manipulation of local filestore)
- **Chapter 8**, **FTAM Profile Details**.

The rest of the document provides background to the specification and discusses some of its key features.

A compliant system shall meet the definitive requirements described in this BSFT CAE Specification.

Trademarks

X/Open™ and the X device are trademarks of the X/Open Company Limited.

IBM® is a registered trademark of International Business Machines Corporation.

UNIX® is a registered trademark of UNIX System Laboratories, Inc. in the USA and other countries.

Referenced Documents

The following documents are referenced in this specification:

FTAM Standards and Profiles

IS 8571-1
: Information Processing Systems - Open Systems Interconnection - File Transfer and Management - General Introduction.

IS 8571-2
: Information Processing Systems - Open Systems Interconnection - File Transfer and Management - The Virtual Filestore.

IS 8571-3
: Information Processing Systems - Open Systems Interconnection - File Transfer and Management - The File Service Definition.

IS 8571-4
: Information Processing Systems - Open Systems Interconnection - File Transfer and Management - The File Protocol Specification.

IS 8571-5
: Information Processing Systems - Open Systems Interconnection - File Transfer and Management - Protocol Conformance Statement Proforma

ISO 8571, WDAD 2 89/01
: File Transfer, Access & Management (FTAM) - Addendum 2: Filestore Management.

ISO/IEC ISP 10607-1:1990
: Information Technology - International Standardized Profiles AFTnn - File Transfer, Access and Management - Part 1: Specification of ACSE, Presentation and Session Protocols for the by FTAM.

ISO/IEC ISP 10607-2:1990
: Information Technology - International Standardized Profiles AFTnn - File Transfer, Access and Management - Part 2: Definition of document types, constraint sets and syntaxes.

ISO/IEC ISP 10607-3:1990
: Information Technology - International Standardized Profiles AFTnn - File Transfer, Access and Management - Part 3: AFT 11 - Simple File Transfer Service (unstructured).

ISO/IEC ISP 10607-6:1990
: Information Technology - International Standardized Profiles AFTnn - File Transfer, Access and Management - Part 6: AFT 3 - Simple File Transfer Service (unstructured)."

ACSE

IS 8649

Information Processing Systems - Open Systems Interconnection - Service Definition for the Association Control Service Element

IS 8650

Information Processing Systems - Open Systems Interconnection - Protocol Specification for the Association Control Service Element

Presentation Layer

IS 8822

Information Processing Systems - Open Systems Interconnection - Connection Oriented Presentation Service Definition

IS 8822

Information Processing Systems - Open Systems Interconnection - Connection Oriented Presentation Protocol Specification

ASN.1

IS 8824

Information Processing Systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1)

IS 8825

Information Processing Systems - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)

Session Layer

IS 8326

Information Processing Systems - Open Systems Interconnection - Basic Connection-Oriented Session Service Definition

ISO 8326 AD2

Information Processing Systems - Open Systems Interconnection - Basic Connection-Oriented Session Service Definition - ADDENDUM 2 - Incorporation of unlimited user data

IS 8327

Information Processing Systems - Open Systems Interconnection - Basic Connection-Oriented Session Protocol Specification

ISO 8327 AD2

Information Processing Systems - Open Systems Interconnection - Basic Connection-Oriented Session Protocol Specification - ADDENDUM 2 - Incorporation of unlimited user data

Transport Service

IS 8072

Information Processing Systems - Open Systems Interconnection - Connection-Oriented Transport Service Definition

FTAM Profiles

US GOSIP
FIPS PUB 146, August 1988

US GOSIP
Draft Version 2.0, April 1989

Government Open Systems Interconnection profile
Users Guide Draft Report

UK GOSIP:
Version 3.0, January 1988.

SPAG Guide to the Uses of standards
Rev 4.0, Spring 1989.

EWOS Documents
EWOS / TA / 88 / 39 (ED 005), (A / 111)"
EWOS / TA / 88 / 36 (ED 006), (A / 112)"
EWOS / TA / 88 / 37 (ED 007), (A / 122)"
EWOS / TA / 88 / 38 (ED 001), (A / 13)"

ENV 21 204, (A / 111)

NIST
Stable Implementation Agreements for OSI Protocols Version 2, December 1989.

MAP / TOP Specification
Version 3.0, February 1989.

Internet FTP Definition

File Transfer Protocol
MIL-STD-1780"

MOD File Transfer Protocol
RFC 959"

POSIX Specification

ISO / IEC 9945-1, 1990 (IEEE Std 1003.1-1990)
Information Technology - Portable Operating System Interface (POSIX) - Part 1: System Application Program Interface (API) [C Language] Institute of Electrical and Electronics Engineers, September 1990.

Transport Service Interface

XTI Specification
Revised XTI (X / Open Transport Interface), CAE Specification, X / Open Company Limited, 1991

Chapter 1

Introduction

BSFT specifies a Byte Stream File Transfer facility using FTAM protocols. BSFT is based on international standards that have reached full IS status, and on internationally harmonised profiles. The FTAM standard consists of the following five parts:

1. General Introduction
2. Virtual Filestore
3. File Service Definition
4. File Protocol Specification
5. FTAM PICS pro forma

All the above parts are full International Standards. The **Referenced Documents** section of this document gives a full list of the base standards and profiles.

This version of the BSFT specification is based on International Standard Profile ISP 10607:1990, File Transfer Access and Management, part 3 (AFT11 - Simple File Transfer Service) and part 6 (AFT3 - File Management Service). In this document, these two specifications are referred to as the *base specifications*.

BSFT supports all mandatory functions and some of the optional functions of the base profiles. This version of the BSFT specification provides the functionality of the Internet FTP utility, using ISO protocol stacks. Therefore, the interface supports only synchronous file transfer and management operations. The BSFT specification aims to make easier the simultaneous use of file transfer facilities in the IPS and OSI environment, and to facilitate the coexistence and migration between these two protocol sets by defining a similar user interface and functionality in the two environments. In selecting FTAM functionality to be included in BSFT, the general principle has been to include all mandatory features of ISP 10607-3 and ISP 10607-6 and to include optional features only if they are required to produce FTP equivalent functionality.

BSFT is primarily intended to support the transfer and management of unstructured files between X/Open-compliant filestores. However, the use of international standards and profiles allows file transfer and management between a BSFT system and any other system that supports the base profiles.

This document does not reproduce the base profiles. Instead it indicates which optional features must be supported and, where appropriate, how the required FTAM functions can be mapped onto X/Open CAE functions. Therefore, implementors must have access to the referenced ISPs.

The core of the BSFT specification lies in the completed PICS, together with the clarifications in **Chapter 9**, **Lower Upper Layers** and in the specification of the user interface in **Appendix A**, **BSFT User Interface Definition**.

Chapter 2

Background

2.1 THE NEED FOR PROFILES

The international standardisation activity in the area of protocols for interworking has now reached a mature stage and many standards have achieved full IS status. However, these standards alone are not sufficient for general interworking because:

- for some layers the OSI specifications define more than one protocol to be met:
  - — different technologies (e.g., ISO 8802-3 for LANs versus X.25 layer 2 for WANs)
  - — different modes of operation (e.g., connectionless and connection-oriented modes of operation at the network layer)
- ISO protocols contain many options (some of which are mutually exclusive) that allow for different levels of functionality
- agreement is needed for some features that are either outside the scope of ISO or for which ISO has not yet reached stable agreement.

The potential interworking problems between OSI implementations of like standards have led to the development of profiles, or functional standards. For a specific requirement (such as file transfer), a profile specifies the base standards and options within the base standards to be used. Profiles generally have a narrower field of application than the base standards. For example, the ISP 10607-3 functional standard covers the transfer of files with no internal structure and supports a restricted set of functions. Different profiles cover the transfer of more complex files.

Profile work was initially driven by bodies who wanted to procure OSI based systems, such as large organisations (e.g., MAP and TOP) and national governments. Later, regional bodies, such as NIST and EWOS, became involved in profiling work. This started to produce an expanding set of incompatible profiles. As interest in profiles increased, so did the realisation that they must be defined with international collaboration if the establishment of incompatible sectoral variants of OSI was to be avoided. ISO is currently co-ordinating the work of three regional bodies (NIST, EWOS and AOW) who are working on internationally harmonised profiles. This work culminates in the definition of International Standardised Profiles (ISPs).

2.2 BASIC ARCHITECTURE OF PROFILES

A fundamental assumption in all profiles is that it is appropriate to ‘‘split’’ the OSI stack at the Transport Service boundary, so that two aspects of interworking can be addressed independently:

- applications interworking (covered by layers 5, 6 and 7)
- network characteristics (covered by layers 1, 2, 3 and 4).

There are a number of profiles that apply to the lower layers (layers 1 to 4) that assume different underlying technologies (e.g., CSMA / CD versus Token Ring) or define different features (e.g., different transport classes). In principle, any set of profiles for the lower layers may support any higher layer application service (layers 5, 6 and 7), although particular lower layer profiles may be more appropriate in particular implementation scenarios.

Irrespective of the profiles selected for layers 1 to 4, the transport layer interface presented to the session layer will be invariant. Provided the minimum service level needed by an application is supported, any variations in the definition of the first four layers should not affect the application requirements. This allows the conformance requirements for layers 1 to 4 to be excluded from the definition of BSFT. ISP*10607-1* implies some basic requirements that must be met by the transport service (these requirements can of course be met by the OSI transport service and protocol).

In contrast to the profiles for layers 1 to 4, the profiles for layers 5 to 7 are specific to each application and each one can be considered independently of the others. Furthermore, for a specified application type, such as file transfer, there are different profiles offering different levels of functionality; these have many common features, including identical requirements for layers 5 and 6.

Initial work in FTAM profiles was carried out by a number of groups, but currently it is being spearheaded by the three regional groups EWOS, NIST and AOW.

These groups are attempting to define mutually agreed profiles for FTAM and lower layers which can be put forward for approval as ISO ISPs. In the case of FTAM, this work has produced a number of profiles, with the simplest being at an advanced stage of development and generally endorsed by all three regional groups. These profiles are prime candidates for ISO ISP status. Work is still needed for the more complex profiles, but the manner in which these will evolve is now clear.

The work of EWOS, NIST and AOW is intended to produce a single profile for each identified level of functionality and specific file type. For instance, there will be a single, universally agreed profile for the transfer of unstructured text files, with no file management functionality. For different functionality, a different profile will be needed (again, universally agreed). Organisations or governments procuring OSI file transfer products will have to decide what functionality they require and select the appropriate profile(s) from those defined by ISO. ISO profiles have a set of mandatory functions and a set of options. Implementations claiming conformance to a particular profile must support all the mandatory features of the profile, and be prepared to propose options and have them negotiated out or to negotiate out optional features they do not support.

2.3 CONFORMANCE TO STANDARDS

There has been increasing interest in placing a more formal definition on conformance. Users would ideally want to be certain that the interworking promised by OSI will be achieved by products they intend to buy. Such assurances can take many forms, such as:

- assurance of the interworking capability of whole products or functions
- assurance of the interworking capability of individual components of a product
- demonstration of interoperability (preferably in the user's environment)
- certification: this generally means that a specific implementation, running in a specific environment, has been shown to pass a set of tests. Users may be more interested in results in plain language and with the precision to support contractual terms
- resolution of interoperability problems between products claiming conformance to the same standards and/or profiles
- performance testing
- world-wide recognition of tests.

Current work aims to express conformance to standards:

- by means of a formal statement of what an implementation must and does support
- by testing an implementation once it has been produced.

ISO work is underway in both areas, with **ISP**s and **PICS**s tackling the former issue; and **ISO 9646**, **Conformance Testing Methodology and Framework**, together with the development of test centres, tackling the latter. This **BSFT Specification** covers only the statement of requirements.

In order to help users, vendors and test centres to specify exactly the functionality of an implementation, ISO has developed **Protocol Implementation Conformance Statements (PICS)**. These are essentially pro formas to allow all options and parameters of standards relevant to an implementation to be specified in a common, concise and unambiguous manner. The base standards define the pro formas to be used, while ISPs include PICS pro forma(s) completed to indicate how the standards should be supported.

2.4 FIELD OF APPLICATION

FTAM services can be used in a number of different ways.

1. The simplest form is that of a file transfer utility providing a screen based user interface to drive the FTAM protocol machine.
2. Alternatively, an application programming interface can give access to FTAM file transfer functionality to other programs. This functionality may be richer than that present at the user interface.
3. Finally, the FTAM functionality can be built into application software, such that the real user would be unaware of the protocol in use. An example would be a printer that receives files by means of the FTAM protocol. Such software may use a programmatic interface to access a standard file transfer module or may itself contain the relevant code.

BSFT defines how FTAM can be used to provide a user driven file transfer utility. It covers the transfer of files to and from an X/Open-compliant filestore and the management of the Virtual Filestore. BSFT does not specify how FTAM will be implemented. It simply defines the functionality that is visible externally to the machine (i.e., when a machine is communicating with another machine using FTAM) and the functionality that is presented to the user.

The use of accepted ISO standards and profiles allows a BSFT system to interwork not only with other BSFT systems but with any other system implementing compatible standards and profiles.

Chapter 3

Overview Of the Base Profiles

3.1 BACKGROUND

ISO profiles are directly related to base standards, and conformance to profiles implies conformance to standards. ISPs are intended to be concise documents which do not repeat the text of the documents to which they refer. Because of the need for common text between related profiles, some ISPs will be produced in a number of separate parts. (An example of a multipart ISP is FTAM.)

The FTAM profiles are multipart because much of the ‘‘Lower Upper Layers’’ (Session, Presentation and ACSE) will be common to all profiles, and hence should be defined in a common document. In addition, ASN.1 is singled out as requiring special treatment since it will be used in all application profiles. It will be desirable for implementations supporting many profiles to use common routines for handling ASN.1, therefore common usage should be encouraged between all profiles.

3.2 ISO FTAM PROFILES

The international standardisation activity for FTAM has identified FTAM profiles that deal with different types of files:

- **Unstructured files**
  These consist of a single File Access Data Unit and therefore can be accessed only as a single unit
- **Flat files**
  These consist of a number of File Access Data Units, all of which are connected to the root File Access Data Unit
- **Hierarchical files**
  These consist of a number of File Access Units that can have the full FTAM hierarchical structure.

ISO have defined profiles that correspond to these three file types. Furthermore, for the Flat and Hierarchical file types, ISO have defined the option of accessing a file at the level of the complete file or of individual File Access Data Units. This gives rise to five profiles. A further profile (intended to be used together with any of the other profiles) adds file management. This results in the following six profiles:

AFT 11	Simple File Transfer (Unstructured)
AFT 12	Positional File Transfer (Flat)
AFT 13	Full File Transfer (Hierarchical)
AFT 22	Positional File Access (Flat)
AFT 23	Full File Access (Hierarchical)
AFT 3	File Management Service.

While there is general agreement between the regional groups on the simpler FTAM profiles (i.e., those dealing with Unstructured and Flat files), the profiles dealing with the Transfer and Access of Hierarchical files have not been developed. This is reflected in the ISP for FTAM which is a multi-part ISP consisting of the following six parts:

Part 1	Specification of ACSE, Presentation and Session Protocols for the use by FTAM
Part 2	Definition of Document Types, Constraint Sets and Syntaxes
Part 3	AFT11 - Simple File Transfer Service (Unstructured)
Part 4	AFT12 - Positional File Transfer Service (Flat)
Part 5	AFT22 - Positional File Access Service (Flat)
Part 6	AFT3 - File Management Service

The requirements of BSFT are covered in parts 1, 2, 3 and 6. Part 6 is only needed in order to allow the modification of ‘‘file attributes’’ (such as filename).

3.3 INTERNATIONAL STANDARDISATION ACTIVITY

The following table summarises the FTAM profiles defined by various profiles groups and indicates how the profiles correspond technically.

ISP 10607	EWOS	NIST	INTAP	US GOSIP	UK GOSIP	CEN/ CENELEC
AFT 11	A / 111	T1	AP.111	T1	GOSIP-A	41 204
AFT 12	A / 112	T2	AP.112	T2		41 206
AFT 13	A / 113	T3				
AFT 22	A / 122	A1	AP.122	A1		41 207
AFT 23	A / 123	A2				
AFT 3	A / 13	M1	AP.12	M1		41 205

Table 3-1: Summary of FTAM Profiles

3.4 STYLE OF ISP 10607-3 AND 10607-6

The ISP 10607-3 profile is a short document, being taken up mostly by a list of detailed requirements expressed in terms of a PICS. This is a subset of the PICS pro forma included in ISO 8571-5 and is intended to give information additional to the base standards. Some ISO 8571-5 PICS sections or subsections are not duplicated in the ISP 10607-3 because the profile does not impose any requirements on them; these sections are designated in the profile as void. The ISP 10607-3 FTAM PICS includes the following sections:

1. Implementation Detail
   This section is implementation-specific and allows the PICS to be related to a specific product. In ISP 10607-3 this section is designated as void (the only complete section so designated).

2. General ISO 8571 Detail
   This covers general details of ISO 8571, such as protocol version number and addenda, and defect reports implemented.

3. Syntax Detail
   This section defines support for abstract and transfer syntaxes. In DIS 8571-5, this section includes only abstract syntaxes required for the basic implementation of the protocol (i.e., not those required for the support of document types). ISP 10607-3 adds to this section the abstract syntaxes required for the support of document types (these are specified in the document definitions included in ISO 8571-2). However, ISP 10607-3 adds the abstract syntaxes for NBS file directory document type (which is not covered in ISO 8571-2).

4. Virtual Filestore Detail
   This section covers the functions of the Virtual Filestore: File Attributes, Constraint Sets, File and Filestore Actions, Access Contexts and Responder Override.

5. File Protocol Detail
   This section comprises the main body of the PICS and defines which protocol features are supported in an implementation. There is a group of entries for each FTAM Protocol Data Unit (PDU) listing all parameters. These are cross-referenced to other sections that give more details for specific PDUs.

6. Document Types
   This section covers the document types supported, including the abstract syntaxes required by these documents. DIS 8571-5 covers the document types defined in IS 8571-2: FTAM1, FTAM2, FTAM3 and FTAM4. The profile adds the NBS-9 FTAM directory file and the INTAP-1 record file (both optional).

The ISO 8571-5 PICS is expressed as a collection of tables each with between two and four columns. These define the following information:

1. Functions and Feature of ISO 8571

2. Initiator Implementation Details
   This has two sub-columns headed D and I. The former indicates whether, according to ISO 8571, the items listed in column 1 are mandatory, optional or inapplicable, or (for attributes) whether full or partial support is required. The latter indicates whether the item in column 1 is required or has been implemented.

3. Responder Implementation Details
   This has two sub-columns headed D and R. The former indicates whether, according to ISO 8571, the items listed in column 1 are mandatory, optional or inapplicable, or (for attributes) whether full or partial support is required. The latter indicates whether the item in column 1 is required or has been implemented.

4. Additional Information
   This column allows additional information to be specified, such as range of values.

In addition to the PICS, the profile contains a short section covering base standards and further details on some of the items in the ISPICS.

ISP 10607-6 has the same style and layout as ISP 106087-3 but adds support for the Enhanced File Management Functional Unit.

Chapter 4

Scope of BSFT

BSFT specifies the transfer and management of files as stored in a X/Open-compliant filestore. FTAM defines an asymmetric protocol between two end-systems, one end acting in the Initiator role and issuing file transfer and management commands and (in the case of BSFT) supporting the command line interface, while the other end-system acts in the Responder role, supporting the FTAM Virtual Filestore (VFS) and (in the case of BSFT) mapping it onto a real X/Open-compliant filestore. The VFS is an abstract model of a filestore that allows FTAM to define a generic filestore and specify how it is acted upon by the FTAM protocol. BSFT defines support for both Responder and Initiator roles and for read and write functionality in both roles. The FTAM model as it applies to BSFT is shown in the figure below.

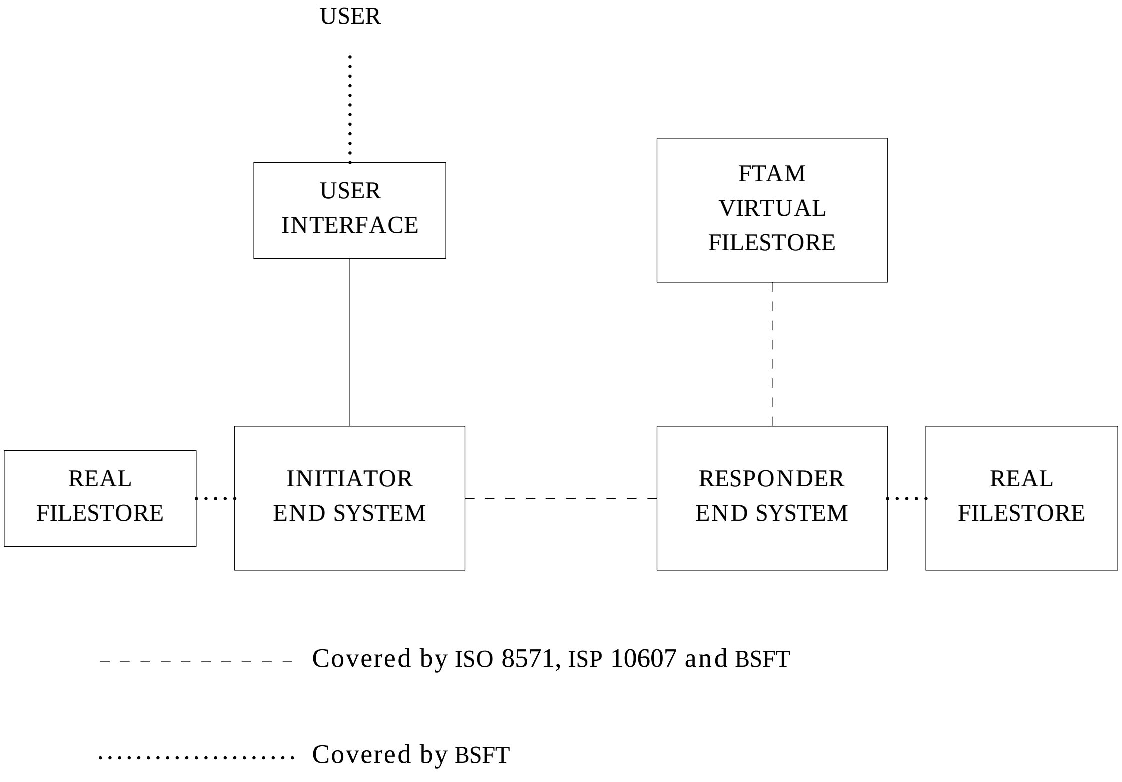


Figure 4-1: Main Components of BSFT

BSFT supports file transfer between X/Open-compliant systems and uses all mandatory functions and some optional functions contained in the base profiles. However, all BSFT implementations will be able to interwork with implementations which only support the mandatory functions (see **Chapter 6**, **BSFT Requirements**). Optional functions may be negotiated out when the FTAM Regime is established or, where allowed by the protocol, ignored. This means that BSFT implementations will be able to interwork with any other implementation (whether it is X/Open-compliant or not), provided that the negotiated functionality is sufficient to their purposes.

As required by the base profiles, BSFT covers the three higher layers of the OSI model:

layer 7	FTAM ACSE
layer 6	Presentation ASN.1 Abstract Syntax ASN.1 Basic Encoding Rules
layer 5	Session Service

BSFT uses the Connection Oriented Transport Service. The **XTI Specification** from X/Open (see **Referenced Documents**) defines an interface through which such a service is available, although there is no requirement for a BSFT implementation to use the XTI Interface itself.

Figure 4-2 shows the FTAM stack as it applies to BSFT. It shows the various options in the upper layers of the FTAM stack. At the FTAM layer various functional units are mandatory in different profiles. Those marked as supported for BSFT are those required for the base profiles. At the transport layer, service primitives are shown since protocol detail is outside the scope of BSFT.

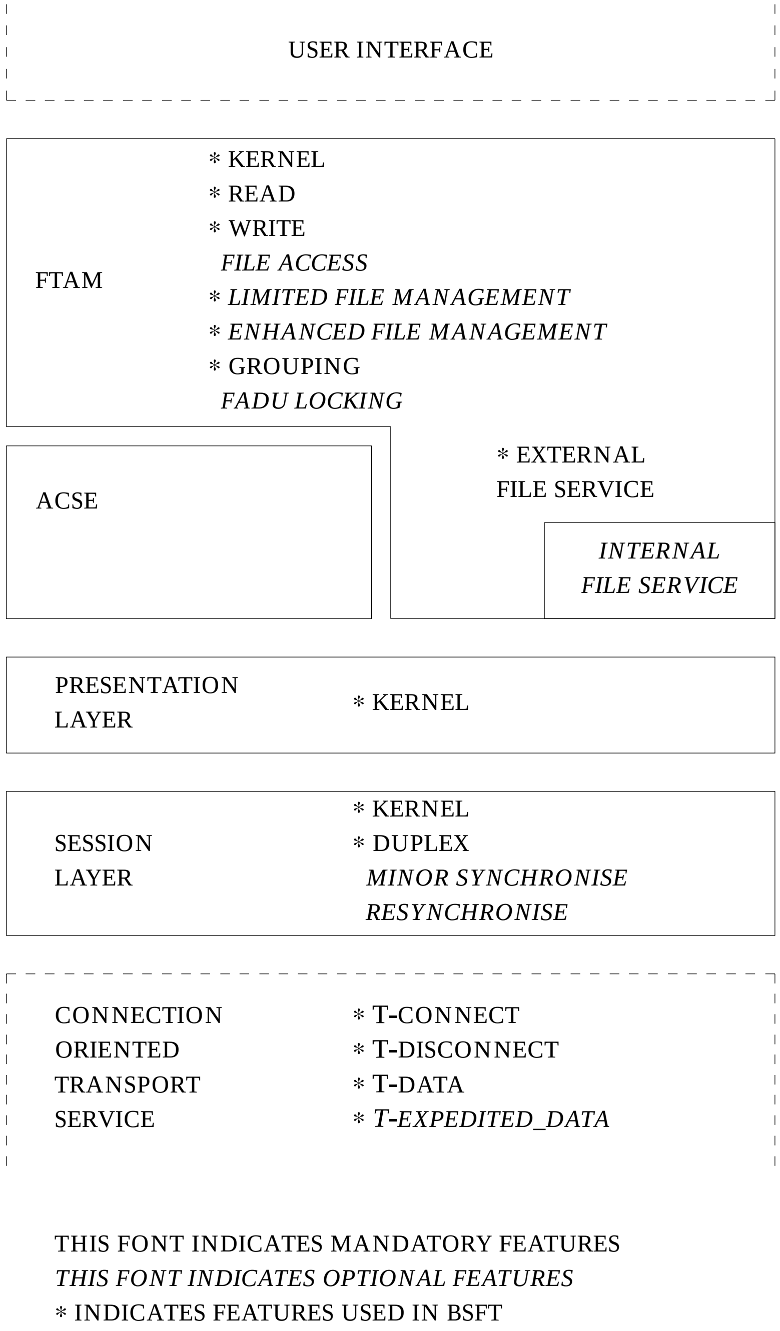


Figure 4-2: Main Components of FTAM Stack

The definition of BSFT can be divided into the following areas:

1. the specification of the layer of protocol and service support required from FTAM, ACSE, Presentation and Session
2. the definition of the command line interface
3. the definition of how the FTAM Virtual Filestore is mapped onto the X/Open-compliant filestore (only aspects that are externally visible are covered, such as the manner in which X/Open CAE filenames are mapped onto the FTAM filename attribute)
4. a statement of the externally visible requirements imposed by BSFT on the Transport Layer.

This document treats the BSFT stack as a single unit. It does not specify the internal structure of the system, nor does it define or depend on any formal internal interfaces. Specifically, it does not define an application programming interface (API) to FTAM, nor does it cover system management functionality or interfaces. Such additional specifications may be added in the future.

BSFT does not define performance or quality of service targets.

The BSFT definition can:

- provide part of a procurement definition
- ensure interoperability between X/Open-compliant systems supporting file transfer based on BSFT
- ensure a uniform command line interface across X/Open-compliant systems supporting file transfer based on BSFT
- allow interoperability of BSFT systems with non-X/Open-compliant systems that implement the Simple File Transfer (Unstructured Files) and File Management profiles
- facilitate the coexistence and migration between the IPS and OSI environments.

Chapter 5

Overview of BSFT

BSFT covers the following two major roles:

1. Initiator of access to a remote filestore (which may be non-X/Open-compliant)
2. Responder, giving access to the local X/Open-compliant filestore to remote Initiators.

The Initiator role is divided into the following lesser roles in order to achieve clarity:

1. Initiator interface support
2. Initiator local file manipulation
3. communications protocol machinery.

The Responder role is similarly divided into the following lesser roles in order to achieve clarity:

1. mapping to local filestore
2. Responder local file manipulation
3. communications protocol machinery.

These roles do not imply a structure for BSFT implementations.

This specification consists of introductory, overview and definitive sections. The definitive sections are:

- **Appendix A**, **BSFT User Interface Definition**
- **Section 7.6**, **File Attributes** through **Section 7.9**, **Invalid Filenames** (mapping to the Responder's filestore)
- **Section 7.9**, **Invalid Filenames** through **Section 7.11**, **Reading Files from a Responder** (Initiator's manipulation of local filestore)
- **Chapter 8**, **FTAM Profile Details**.

BSFT supports the Transfer and Transfer & Management Service Classes in both Initiator and Responder roles. In the Responder role, BSFT also supports the Management Service Class. The Restart and Recovery Functional Units (which are optional in these Service Classes) are outside the scope of the BSFT specification. BSFT normally attempts to negotiate the Transfer & Management Service Class but also allows the Transfer Service Class to be negotiated. BSFT supports the Kernel (mandatory) and Storage (optional) attribute groups (some Storage Group Attributes are partially supported). Since BSFT does not require support for the Security attribute group some files on some hosts may not be accessible.

BSFT supports both Text and Binary files. In accordance with ISP 10607, these files are assumed to have no internal structure that is visible to the file transfer system. Files received by BSFT that are designated as text files by the FTAM protocols are translated to X/Open-compliant text files. Similarly, files received by BSFT that are designated as binary files by the FTAM protocols are translated to X/Open-compliant 8-bit binary files. BSFT does not define how the Text/Binary distinction may by preserved once files are written to the X/Open-compliant filestore.

Access to directories is partially supported. ISO is developing directory support for FTAM in the **FTAM Filestore Management** Addendum to ISO 8571. This is currently at the working draft stage and it is not supported by any of the profile groups - hence it is not supported by BSFT (it may be supported in the future if adopted by the profile groups). BSFT allows full pathnames to be specified and directory information to be read from Responders. To support the transfer of directory information and to implement wildcard file specifications, BSFT supports the NBS-9 document type (optional in ISP 10607-3). Although Responders do not explicitly recognise file links, the creation and deletion of directories are outside the scope of the BSFT specification.

As the Restart and Recovery Functional Units are outside the scope of the BSFT specification, BSFT conformance does not require the support of any error recovery. However, file transfer operations will benefit from any error detection/correction procedures supported by the Transport Service provider.

As a general rule, ISP 10607 options that are not directory visible at the user interface (such as Restart or Session Segmentation) are outside the scope of the BSFT specification.

In the completed PICS included in **Chapter 8**, **FTAM Profile Details** of this document, all features that are mandatory in BSFT but not in ISP 10607 have been clearly marked. See also **Chapter 3.4**, **Style of ISP 10607-3 and 10607-6** for a description of how the PICS is expressed.

Chapter 6

BSFT Requirements

To conform to the BSFT specification an implementation will:

1. support the FTAM functionality as defined in ISP 10607-3, the completed PICS in the BSFT specification and **Chapter 8**, **FTAM Profile Details**

2. support ISP 10607-2 profile as it applies to the NBS-9 file document type.

3. support the ACSE, Presentation and Session specification of ISP 10607-1 with the qualifications in **Chapter 9**, **Lower Upper Layers**

4. support the user interface defined in **Appendix A**, **BSFT User Interface Definition**. Where a user requests a function that requires a feature that is optional in the base profiles and has been negotiated out, then an error message will be produced that clearly indicates that the requested function is normally valid but is currently not available.

5. support the mapping between FTAM and X/Open-compliant filestore defined in **Chapter 7**, **Technical Requirements**. **Chapter 8**, **FTAM Profile Details**, and ISO 8571 place limits on the range and/or type of values permitted for parameters involved in this mapping.

6. be capable of requesting all features that are specified as mandatory in BSFT during the establishment of the FTAM Regime (even though some of these features are specified as optional in the base profiles). Features that are specified as mandatory in BSFT but which are specified as optional in the base profiles may be rejected (negotiated out or ignored as appropriate) by a Responder which does not conform to BSFT.

A BSFT Responder may reject any features which are specified as optional in the BSFT specification and which are proposed by the Initiator but are not supported by the Responder.

Non-negotiable features that are optional in the base profiles may be absent in a non-BSFT Initiator or a non-BSFT Responder. BSFT implementations will continue to interwork with such implementations.

Chapter 7

Technical Requirements

7.1 INTRODUCTION

This chapter defines the interaction between a BSFT Initiator and an FTAM Responder, and between a BSFT implementation and an X/Open compliant filestore, in terms of the FTAM protocol.

An FTAM session is divided into a number of phases, which are termed Regimes. The following Regimes are defined:

- FTAM Regime
- File Selection Regime
- File Open Regime
- Data Transfer Regime.

An FTAM Regime must first be established. The Initiator provides the information necessary to address the remote system, identify itself, select the necessary quality of service and select the required presentation contexts. The Responder then performs any necessary authentication of the Initiator and (if acceptable) confirms its acceptance of the association.

A file in the Responder may then be *Selected* or *Created*; this establishes the File Selection Regime. When the file is *Selected* or *Created*, a number of parameters can be specified that define the type of access and concurrency control required on the file, passwords required by the Responder for the access requested by the Initiator, and an account that applies to the File Selection Regime. This is followed by the File Open Regime, in which the file is opened for reading or writing. Only one read or write is permitted in the File Open Regime. When this is complete, the file is *Closed* and *Deselected* or *Deleted*. If no further file transfer or management activity is required, the FTAM Regime is terminated.

FTAM allows a number of File Selection Regimes to be serially established within a single FTAM Regime.

7.2 ADDRESSING AND RELATED ISSUES

The FTAM stack requires addressing and related parameters to be specified both at the FTAM layer and at the lower layers of the FTAM stack. These are summarised in Table 7-1 below. The FTAM standard and profiles specify the semantics of the FTAM parameters and define values for the ACSE parameters. Lower layer addressing parameters are outside the scope of ISO 8571 and the base profiles. BSFT implementations support a flexible addressing scheme for the lower layers to allow addresses for layers at and below the Presentation layer to be compatible with other communications facilities of the host machine and to comply with the addressing scheme of the communications environment in which they will be used.

STANDARD	PARAMETER	TYPE
FTAM	Initiator Identity	GraphicString
	Account	GraphicString
	Filestore Password	GraphicString or OCTET STRING
ACSE	Application Context Name	OBJECT IDENTIFIER
	Application Process Title	OBJECT IDENTIFIER
	Application Entity Qualifier	INTEGER
Presentation	Presentation Selector	OCTETSTRING
Session	SSAP Identifier	string of octets
Transport	TSAP Identifier	string of octets
Network	NSAP Identifier	string of octets

Table 7-1: Addressing and Related Parameters in the FTAM Stack

The FTAM addressing parameters in Table 7-1 above are intended to be used as follows:

- **Initiator Identity**
  This parameter identifies the calling user. In BSFT Initiators, values for this parameter are specified via the user interface and are treated as a variable length string. BSFT does not specify what action, if any, BSFT Responders will take on receipt of this parameter. A default value may be generated by a BSFT Initiator in an implementation defined manner.

- **Account**
  This parameter identifies the account to which costs incurred by the regime will be charged. FTAM allows separate ‘‘account’’ parameters to be specified for the FTAM Association and File Selection Regimes. In BSFT Initiators, values for the FTAM Association Regime account (carried in the *F-INITIALISE PDU*, when the association is established) are specified via the user interface and are treated as a variable length string. The support of the File Selection Account parameter is outside the scope of BSFT. A default value may be generated in an implementation defined manner.

Of the ACSE parameters in the table above, a constant value is defined for the Application Process Name (in ISO 8571).

The Presentation Selector, together with the Session Selector, Transport Selector and NSAP, are generated by local means from information supplied by the user of the BSFT command line interface.

7.3 PASSWORDS

Passwords appear in a number of places in FTAM, not all of which are mandatory, nor can they all be mapped onto X/Open CAE facilities.

- **Filestore Password**
  A password is present in the *F-INITIALISE* and applies to the entire filestore. It is of type **GraphicString** or **OCTET STRING** and is mandatory in the base profiles for Initiators and optional for Responders. In BSFT Initiators, values for this parameter are specified via the user interface and are treated as a variable length string. BSFT does not specify what action, if any, BSFT Responders will take on receipt of this parameter (it may be mapped onto an X/Open-compliant login password).

- **Create Password**
  FTAM supports a ‘‘create’’ password that is quoted in the *F-CREATE* and determines if an Initiator is allowed to create a file in the current directory. It is of type **GraphicString** or **OCTET STRING** and is mandatory in the base profiles for Initiators and optional for Responders. This is supported by BSFT Initiators. BSFT Responders will accept any value.

- **Access Passwords**
  FTAM also supports passwords applying to a specific operation (such as read, extend, delete, etc.) on a single file. These are used in the *F-SELECT* and *F-CREATE* (if an existing file is selected) and are of type **GraphicString** or **OCTET STRING**. The access passwords are available only if the Security Attribute Group is supported, which is optional in the base profiles, and therefore an implementation conforming to these profiles must be prepared to handle the case where the Access Passwords attribute is absent. BSFT does not support the Security Group and hence does not support Access Passwords.

A further degree of protection is enforced by the ‘‘permitted actions’’ FTAM attribute. This is set when the file is created and defines which actions are allowed on a specific file. BSFT supports those parameters that are appropriate to unstructured files (i.e., all parameters except *insert* and *erase*, which require files with internal structure). The permitted actions attribute is a Boolean vector with the elements listed in Table 7-2 below (‘‘change attribute’’ requires the File Management Service profile). These elements will be set according to the file’s X/Open CAE access rights.

Permitted Actions on Created Files

When creating files, BSFT Initiators should set the following Permitted-Actions attribute values as TRUE for the F_CREATE Initial-Attributes parameter:

read, replace, extend, read-attributes, delete-file.

ELEMENT	VALUES SUPPORTED BY BSFT
Actions Available	
read	TRUE or FALSE
insert	FALSE
replace	TRUE or FALSE
extend	TRUE or FALSE
read attribute	TRUE or FALSE
delete file	TRUE or FALSE
change attributes	TRUE or FALSE
erase	FALSE
FADU-Identity Groups Available	
traversal	TRUE or FALSE
reverse traversal	TRUE or FALSE
random order	FALSE

Table 7-2: Permitted File Actions Supported by FTAM

7.4 FILE ACTIONS

FTAM defines a number of actions on files (‘‘actions on complete files’’), and, once a file has been selected, a number of actions on selected files (‘‘actions for access on files’’). BSFT supports all actions on complete files defined by FTAM, and actions for access on files that are appropriate to unstructured files (defined as mandatory in the base profiles). The File Management Service profile is required for the ‘‘change attribute’’ action. The level of BSFT support of these features in both Initiator and Responder roles is summarised in Table 7-3:

Actions on Complete Files				
	BSFT	**FTAM**	**ISP 10607-3**	**ISP 10607-6**
create file	m	o	o	m
select file	m	m	m	m
change attribute	m	o	—	m
read attribute	m	o	o	m
open file	m	m	m	—
close file	m	m	m	—
delete file	m	o	o	m
deselect file	m	m	m	m
Actions for Access on Files				
	BSFT	**FTAM**	**ISP 10607-3**	**ISP 10607-6**
locate	/	—	—	—
read	m	o	m	—
insert	/	—	—	—
replace	m	o	m	—
extend	m	o	m	—
erase	/	o	|	—

m	mandatory
o	optional
—	non-meaningful - not defined in this context
|	outside the scope of ISP 10607-x
/	outside the scope of BSFT

Table 7-3: File Actions Supported by FTAM

7.5 FILE TYPES

ISP 10607-3 supports only unstructured binary and unstructured text files. FTAM allows the contents type of a file to be identified either by specifying a document type or by specifying a constraint-set and abstract-syntax. ISP 10607-3 uses the former method and supports the following file document types:

FTAM-1:	for unstructured text files
FTAM-3:	for unstructured binary files.

The data types of FTAM-1 files are limited to the ASN.1 types **IA5String**, **GraphicString**, **VisibleString** and **GeneralString** (a subset of the types defined in ISO 8571-2 for the FTAM-1 document type). BSFT will support both FTAM-1 and FTAM-3 in both Responder and Initiator roles. When encoding unstructured text data, any ASN.1 type permitted by ISP 10607-3 may be used.

When working as an Initiator and sending a file, BSFT allows the user at the terminal to state the document type and, for FTAM-1 files, the data type to be used. If a user does not specify a document type, the BSFT Initiator determines the document type of files transmitted. When receiving files, BSFT displays to the user's terminal the document type and, where applicable, the data type of received files.

When working as a Responder, BSFT determines the contents type of files read by the Initiator. This is used to verify the proposed contents type in *F-OPEN* if this parameter is not set to ‘‘unknown’’.

The manner in which the contents type of files is determined is implementation specific. For example, the UNIX *file* command may be used.

In addition to FTAM-1 and FTAM-3, BSFT supports the NBS-9 document type. This is an optional document type in the base profiles and is intended to allow directory information to be read from a Responder; it is covered in more detail in **Section 7.7**, **Directory Support**.

BSFT supports the string length and string significance parameters as described in **Sections 6.4**, **A.13.1.2** and **A.13.3.1** of **ISP 10607-3**. The table defining the support level for combinations of ‘‘universal-class-number’’ and ‘‘string-significance’’ parameters is reproduced below.

		String significance		
	universal class number	**variable**	**fixed**	**not-significant**
26	Visible String	m	m	/
22	IA5 String	/	/	m
25	Graphic String	m	m	/
27	General String	/	/	m

m	mandatory
/	outside the scope of BSFT

Table 7-4: BSFT Support of String Significance

7.6 FILE ATTRIBUTES

FTAM specifies a number of parameters that apply to specific files; these are termed file attributes. FTAM divides the file attributes into four Attribute Groups:

- Kernel Group
- Storage Group
- Security Group
- Private Group

and specifies that a Group is supported if all attributes in the Group are either fully supported (a meaningful value is returned) or partially supported (*no-value-available* is returned).

BSFT fully supports the Kernel Attribute Group and partially supports the Storage Attribute Group (i.e., BSFT returns *no-value-available* for some of the attributes in the Storage Group). However, a BSFT Initiator will accept meaningful values for parameters for which a BSFT Responder would return *no-value-available*. It is an implementation decision whether a BSFT Initiator presents such extra information in the Kernel and Storage groups, provided the same action is always followed. The Security and Private Groups are not supported by BSFT.

Below is a list of the attributes in the Kernel and Storage Groups, together with a description of a BSFT Responder's support for these attributes. (A BSFT Initiator always supports meaningful values for these parameters; this is mandated by ISP 10607-3.)

Kernel Group

The following are all supported by BSFT Responders:

- **Filename**
  See **Section 7.8**, **Filename Support**
- **Permitted Actions**
  See **Section7.3**, **Passwords**
- **Contents Type**
  See **Section 7.5**, **File Types**.

Storage Group

- **Storage Account**
  This attribute identifies the accountable authority responsible for accumulated file storage charges. Support of this attribute is not required for BSFT Responders.
- **Date and Time of Creation**
  X/Open-compliant systems do not maintain the date and time of creation of files. BSFT Responders may return *no-value-available*.
- **Date and Time of Last Modification**
  This is mapped onto the corresponding X/Open CAE parameter by BSFT Responders.

- **Date and Time of Last Read Access**
  This is mapped onto the corresponding X/Open CAE parameter by BSFT Responders.
- **Date and Time of Last Attribute Modification**
  This is mapped onto the corresponding X/Open CAE parameter by BSFT Responders.
- **Identity of Creator**
  This is mapped onto the X/Open CAE file owner parameter by BSFT Responders.
- **Identify of Last Modifier**
  X/Open-compliant systems do not have an equivalent parameter. BSFT Responders may return *no-value-available*.
- **Identity of Last Reader**
  X/Open-compliant systems do not have an equivalent parameter. BSFT Responders may return *no-value-available*.
- **Identity of Last Attribute Modifier**
  X/Open-compliant systems do not have an equivalent parameter. BSFT Responders may return *no-value-available*.
- **File Availability**
  This can be either ‘‘immediate availability’’ or ‘‘deferred availability’’. BSFT Responders may fully support this parameter and use it, for example, to indicate that a file is stored on a demountable device.
- **Filesize**
  This is mapped onto the corresponding X/Open CAE parameter by BSFT Responders.
- **Future Filesize**
  This indicates the size to which a file may grow. This parameter may be either fully or partially supported. It may, for example, be used to limit the maximum size of a file. When a ‘‘filesize’’ reaches the ‘‘future filesize’’, the Responder may simply increase its value and issue a warning to the Initiator, or not increase its value and indicate an error to the Initiator.

7.7 DIRECTORY SUPPORT

An area where the FTAM standard is currently deficient is directory management. Work is underway to define directory support in FTAM. This is covered in a **Filestore Management Addendum** to ISO 8571, but this document is currently at working draft stage and is not referenced by any of the profiles. BSFT does not currently use any of the functions defined in the Filestore Management Addendum, although these may be included in the future when they become stable and incorporated into FTAM profiles.

The absence of directory support is generally seen as an important deficiency, and to partly overcome it NIST have defined a document type (NBS-9) that is intended to allow an Initiator to read filestore directory information from a Responder. NBS-9 has also been adopted by EWOS, AOW and ISP 10607. The definition of this document type is included in ISP 10607-2 and is generally recognised as existing only for an interim basis, until ISO complete the FTAM addendum for Filestore Management.

Since NBS-9 is optional in the base profiles, BSFT Initiators must be prepared to interwork with Responders that do not support this file type. The absence of support for NBS-9 will result in a number of functions not being available at the BSFT Initiator user interface. These functions include not only directory listing (the *dir* command in **Section 7.13**, **Working Directory**) but also all functions that allow wildcards to be specified in file names on the Responder.

NBS-9 consists of one File Access Data Unit (FADU), which comprises zero, one or more Data Elements (DEs), each being a single directory entry. For conformance to the BSFT specification it is only necessary to read this document type. Explicitly creating, deleting, writing or modifying this document type is outside the scope of BSFT. The general structure of NBS-9 is defined in ISP 10607-2. However, the parameters that can be included in an NBS-9 document are those defined in ISO 8571 as the ‘‘file attributes’’. FTAM allows directory information for a file to be transferred as parameters for specific file actions, whereas NBS-9 allows directory information for a collection of files to be transferred as a document. An FTAM implementation must map real file directory information onto the parameters of NBS-9. NBS-9 defines the parameters listed in Table 7-5. An I indicates that the parameter will be displayed by a BSFT Initiator (if transmitted by the Responder), while an R indicates that the parameter will be fully supported by a BSFT Responder. The support of parameters not marked I or R is outside the scope of the BSFT specification.

BSFT Support		**Parameter**
I	R	contents type
I	R	storage account date and time of creation
	R	date and time of last modification
	R	date and time of last read access date and time of last attribute modification
I	R	identity of creator identity of last modifier identity of last reader identity of last attribute modifier file availability
I	R	file size future filesize
I	R	access control legal qualifications private use

Table 7-5: Filestore Directory Information (NBS-9) Supported by FTAM

For conformance to NBS-9, a Responder is only required to return the filename attribute, subject to local security and access control. Therefore, BSFT implementations will be prepared to accept associations where the NBS-9 document type cannot be used, or where the information transferred by NBS-9 documents is less or more than information supported by BSFT. When a received NBS-9 document contains more information than that supported by a BSFT Initiator, the amount and format of additional information displayed is implementation-defined.

The Object Identifier value to be used for the NBS-9 document type is currently under review. The BSFT specification will be updated to reflect the decision taken by ISO on this issue.

7.8 FILENAME SUPPORT

The FTAM standard expresses filenames in terms of a ‘‘Filename Attribute’’. The filename attribute defined in ISO 8571 is a vector attribute, the elements of which are of type **GraphicString**. Directories are not defined in the standard at present but are being addressed in the Filestore Management Addendum to ISO 8571.

The mapping of the FTAM Filename attribute onto real filestore filenames is outside the scope of ISO 8571 and the FTAM profiles. The base profiles state that file names shall be specified in the naming convention of the responding FTAM implementation (subject to the requirements of ISO 8571-2).

In conformance to ISP 10607-3, BSFT use ssingle **GraphicString** filenames; that is, X/Open CAE pathnames/filenames are encoded as a single variable length string of type **GraphicString**.

The BSFT interface allows ‘‘wildcards’’ to be specified in some of the commands described in **Appendix A**, **BSFT User Interface Definition**. It should be noted that in all such cases wildcards are expanded in the Initiator. The NBS-9 document type is used when ‘‘wildcards’’ apply to the Responder filestore.

7.9 INVALID FILENAMES

The Initiator role of BSFT (and its user interface) accepts non-X/Open-compliant filenames, because the Responder system need not be X/Open compliant. The BSFT interface allows a user to supply one filename, and BSFT uses derivatives of that filename in both the local filestore and remote filestore. The rules for these situations are as follows:

- the Responder is responsible for checking the filename communicated to it. The Initiator does not attempt to apply any checks to names it is sending;
- if the user does not supply a local filename, and the remote filename cannot be used in the local store, (i.e., it is not a valid X/Open-compliant filename), BSFT will use a local implementation-dependent mechanism to produce a valid local filename. One such mechanism is the *tmpnam* command;
- if the user does not supply a remote filename, then the BSFT Initiator passes the local filename to be used as the remote filename.

7.10 WRITING FILES TO A RESPONDER

When BSFT is acting as an Initiator and is transferring a file to a Responder its first action is to *Create* the file (for *put*/*mput* commands in **Appendix A**, **BSFT User Interface Definition**) or *Select* it (for *append* command in **Appendix A**) and then *Open* it for writing. When creating a file, FTAM defines an ‘‘override’’ parameter that indicates the action to be taken if the file already exists. FTAM defines the following ‘‘override’’ values:

1. *create-failure* will fail the create action if the file already exists. This is the FTAM protocol default and is mapped from the BSFT ‘‘override fail’’ command value.
2. *select-old-file* will select the file if it already exists. This is mapped from the BSFT *append* command.
3. *delete-and-create-with-old-attributes* will delete the file if it already exists and create a new file using the old file’s attributes (effectively delete the contents of the existing file and select it). Support of this value is not required for BSFT conformance.
4. *delete-and-create-with-new-attributes* will delete a file if it already exists and create a new file using new attributes. This will be the BSFT default, mapped from the BSFT *override overwrite* command value.

The last three options may require the specification of an access password (in addition to the optional create password). The access password can be quoted only if the Security Attribute Group is supported. The Security Group is optional and is outside the scope of BSFT.

7.11 READING FILES FROM A RESPONDER

BSFT writes received files to the local X/Open-compliant filestore when acting as an Initiator and reading files from a Responder. The name to be used for the local file is either the local filename quoted in command *get* (see **Section 7.10**, **Writing Files to a Responder**) or, if this is not specified, the remote filename. The use of the latter as quoted in the *get* command may lead to two potential problems:

- filename clashes
- invalid filenames.

The local override parameter is used to resolve filename clashes, i.e., this parameter determines whether files in the local filestore will be overwritten or cause the transfer to fail. Under these circumstances the default value is again *overwrite* (as in **Writing Files to a Responder**, see **Section 7.10**), although other values may be specified by the user.

The same local override parameter is used to set both the default FTAM override parameter and in determining the action to be taken with clashing file names when receiving files as an Initiator.

Invalid filenames are resolved as described in **Section 7.9**, **Invalid Filenames**.

7.12 RESPONDER IMPLEMENTATION OF FTAM VFS

For compliance with the FTAM standard and profiles, BSFT must support the VFS. This defines exactly the behaviour BSFT must exhibit, e.g., filename clashes must be resolved using the override parameter subject to local protection rights.

7.13 WORKING DIRECTORY

The currently stable FTAM specification does not have the concept of a ‘‘working directory’’ (this concept is defined in the File Management Addendum to ISO 8571). A working directory facility can still be supported, provided it is implemented entirely in the Initiator. BSFT supports such a facility.

BSFT implementations acting as Initiators maintain a local variable holding a remote ‘‘working directory’’, to which the filename specified by the local user is appended in order to produce an FTAM filename attribute which is transmitted to the Responder. The working directory parameter is encoded as described in **Section 7.8**, **Filename Support**.

7.14 CHARGING DETAILS

The FTAM protocol allows charging information to be returned by the Responder to the Initiator to indicate costs attributed to an account during the FTAM Association Regime or (if the initial account is overridden in F-SELECT) during the File Selection Regime. This is an optional facility in the base profiles and is outside the scope of BSFT.

7.15 ERROR MESSAGES

A number of FTAM protocol data units contain a diagnostic field that allows the following information to be returned to the Initiator:

1. **Diagnostic-Type**
   This can take the values *permanent*, *transient* or *informative*.

2. **Error-Identifier**
   This categorises the error in terms of the protocol. ISO 8571-3 defines a range of values.

3. **Error-Observer**
   This indicates the type of entity that detected the error. This can indicate Initiator/Responder user or protocol machine, underlying service supporting FTAM, or no categorisation possible.

4. **Error-Source**
   This identifies the entity perceived as the source of the error. This can indicate Initiator user, Responder user, Initiator protocol machine, Responder protocol machine, underlying service supporting FTAM, or no categorisation possible.

5. **Further-Details**
   This is an optional field that contains a natural language message giving additional details on the cause of the error. It may, for example, be displayed at the user terminal in the Initiator system.

When working as a Responder, BSFT may use the optional Text Message field to indicate causes of errors. When working as an Initiator, BSFT displays received Text Message fields; it may also generate additional text error messages based on the other diagnostic fields or on local knowledge.

When a user requests a function that requires a facility that is optional in the base profiles and has been negotiated out by the peer system, BSFT produces an error message that indicates that the requested function is normally available but is currently not available.

7.16 CONCURRENT ACCESS TO FILES

When selecting a file, FTAM allows the Initiator to define a concurrency control parameter. If required by the Initiator this can be further restricted when a selected file is opened. The concurrency parameter is a vector that specifies for each requested access on a file (defined in the Requested Access Parameter) the type of access lock requested by the Initiator. The locks define the access required by the Initiator and the access available to any other user. Table 7-6 summarises which locks are defined:

LOCKS	I MAY PERFORM THE ACTION	OTHERS MAY PERFORM THE ACTION
no access	NO	NO
not required	NO	YES
shared	YES	YES
exclusive	YES	NO

Table 7-6: Concurrency Locks Supported by FTAM

BSFT Initiators supports the concurrency parameter in the *F-SELECT*, *F-CREATE* and *F-OPEN* commands. BSFT Responders operate a default Concurrency Attribute as defined in ISP 10607-3, with the exception that shared access is used when no write actions (*replace, extend*, *delete*, *change attributes*) have been requested (ISP 10607-3 allows, in addition, exclusive access as a local option). ISP 10607-3 defines locks to be used, depending on whether any write action has been requested, as summarised in Table 7-7.

ACTION		ANY WRITE ACTION REQUESTED[1]	
		YES	NO
Read	Requested	shared	shared
	Not Requested	no access	not required
Read Attribute	Requested	shared	shared
	Not Requested	no access	not required
Change Attribute	Requested	exclusive	—
	Not Requested	no access	no access
Replace	Requested	exclusive	—
	Not Requested	no access	no access
Extend	Requested	exclusive	—
	Not Requested	no access	no access
Delete File	Requested	exclusive	—
	Not Requested	no access	no access

[1] Write Actions are *Replace, Extend, Delete File, Change Attributes.*

Table 7-7: Concurrency Locks used by BSFT

7.17 ERROR PROCEDURES

BSFT does not require the support of any error recovery procedures above layer 4. This means that support of the FTAM Recovery and Restart Functional Units (optional in the base profiles) is not required.

BSFT will benefit from error recovery if present at the Transport Layer (which would require an error recovery class such as 1 or 3 or the error detection and recovery features of class 4) or lower layers.

7.18 IMPLEMENTATION INFORMATION

FTAM allows an *Implementation Information* parameter in *F_INITIALISE* that is intended to be used to distinguish between implementations of a specific version number on different equipment. BSFT does not require the use of this parameter when acting as an Initiator. If present in an *F-INITIALISE* when acting as a Responder, as an implementation option, it may be discarded (it will not cause the *F-INITIALISE* to be rejected).

7.19 F-CANCEL MAPPING

The support of the Minor Synchronisation and Resynchronisation Session and Presentation Functional Units is implementation-defined. However, BSFT Responders must be prepared to interwork without the Minor Synchronisation and Resynchronisation Session and Presentation Functional Units; in such cases the *F-CANCEL* is mapped onto the *P-DATA* (as required by ISO 8571).

7.20 SUPPORT OF SERVICE CLASSES

Conformance to the BSFT specification requires the following Service classes to be supported:

BSFT Initiators	Transfer Class Transfer and Management Class
BSFT Responders	Transfer Class Management Class Transfer and Management Class

BSFT Initiators are not required to support the Management Service Class because when the FTAM Regime is established they have no means of knowing the functionality required by the user, therefore they must try to negotiate the highest functionality Service Class supported by the base profiles, namely the Transfer and Management Class. However, since the Transfer and Management Service Class is outside the scope of ISP 10607-3, BSFT Initiators must be prepared for the Responder to negotiate the Transfer Class.

7.21 SUPPORT OF DEFECT REPORTS AND AMENDMENTS

OSI has a procedure whereby defect reports can be raised for ISO standards. If such defects become accepted they are issued as ‘‘Amendments’’ to the standards. The support of Amendments to the FTAM base standards is indicated in **Chapter 8**, **FTAM Profile Details**, **Section 8.2.2**, **Sub-section A.5**, **Defect Report Numbers and Amendments Implemented**.

Chapter 8

FTAM Profile Details

8.1 INTRODUCTION

To specify the FTAM requirements definitively, an FTAM PICS *proforma* must be completed. The FTAM PICS is defined in part 5 of ISO 8571. It can be used by a user or purchaser to define his requirements and by a vendor or implementor to specify which parameters, attributes or other details have or have not been implemented. The base profiles specify some of the PICS options in ISO 8571-5, but still contain a number of profile options. This **BSFT Specification** includes a completed PICS to indicate the FTAM requirements. Items marked ‘‘m’’ in the left-hand-side are mandatory in either of the base profiles.

A completed PICS is intended to be accompanied by comments to clarify some of the selected options. These comments can be found in the base profiles and in **Chapter 9**, **Lower Upper Layers**.

In completing the PICS, the convention describing the support level of protocol features is as defined below:

Y	these features must be supported for conformance to the BSFT specification;
N	these features must not be support in a BSFT implementation;
/	these features are outside the scope of the BSFT specification; however, the support of these features in BSFT Responders must not prevent their interworking with BSFT Initiators;
—	these features are inapplicable.

In tables allowing partial and full support of FTAM parameters, a ‘‘Y’’ under partial indicates the minimum level of support acceptable.

8.2 PICS

Annex A (Normative)
Protocol Implementation Conformance Statement (PICS) Pro forma
for
OSI File Transfer, Access and Management (FTAM)

8.2.1 Section One: Implementation Details

Note that an ‘‘m’’ on the left of a table entry indicates that the value indicated for corresponding function, parameter, etc., is mandatory in one of the base profiles.

A.1 Date of Statement

1	Date of statement yy-mm-dd

A.2 Implementation Details

Specify the information necessary to uniquely identify the implementation and the systems in which it may reside. This may include details of:

a. supplier, implementation name, operating system, suitable hardware;

b. system supplier and/or client of the test laboratory that is to test the implementation;

c. information on whom to contact if there are queries concerning the content of this PICS;

d. the relationship between this PICS and the System Conformance Statement for the systems (see note 1);

e. profiles to which conformance is claimed (see note 2).

NOTES

1. *The System Conformance Statement is defined in ISO 9646. It relates to a PICS covering more than one layer of the reference model.*

2. *The list of profile names is not necessarily a fully inclusive set of those covered by this implementation.*

1

8.2.2 Section Two: General ISO 8571 Detail

A.3 ISO 8571 Protocol Versions Implementation

1m	FTAM protocol version numbers	One

A.4 ISO 8571 Addenda Implemented

1m	ISO 8571-1	—
2m	ISO 8571-2	—
3m	ISO 8571-3	—
4m	ISO 8571-4	—
5m	ISO 8571-5	—

A.5 Defect Report Numbers and Amendments Implemented

The numbers of any approved defect reports and amendments which have been implemented shall be stated below.

1m	ISO 8571-1	ISO 8571-1 / Cor.1:1990
2m	ISO 8571-2	ISO 8571-2 / Cor.1:1990
3m	ISO 8571-3	ISO 8571-3 / Cor.1:1990
4m	ISO 8571-4	ISO 8571-4 / Cor.1:1990
5m	ISO 8571-5	—
	ISO 8650	ISO 8650 / AM1:1989
5m	ISO 8327	8326 / 002: ISO / IEC JTC1 SC21 N 4659 8326 / 005: ISO / IEC JTC1 SC21 N 4660 8326 / 025: ISO / IEC JTC1 SC21 N 4637 8326 / 026: ISO / IEC JTC1 SC21 N 4638 8327 / 037: ISO / IEC JTC1 SC21 N 4661 8327 / 043: ISO / IEC JTC1 SC21 N 4663 8327 / 045: ISO / IEC JTC1 SC21 N 4664 8327 / 047: ISO / IEC JTC1 SC21 N 4665 8327 / 048: ISO / IEC JTC1 SC21 N 4665

A.6 Global Statement of Conformance

1m	Does the implementation referred to in this PICS conform to ISO 8571? yes or no	Yes

A.7 Initiator Responder Capability

State which combination of roles are, and which are not, implemented and specified in this PICS.

	ROLES	D	I	R
1	Sender	o	Y	Y
2	Receiver	o	Y	Y

A.8 Application Context Name Details

List the names of the Application Context Names State recognised or provided by this implementation.

1m	ISO 8571-4 defines a value for a simple transfer mechanism. Other values are not defined in ISO / IEC ISP 10607-3.

8.2.3 Section Three: Syntax Detail

A.9 Abstract Syntaxes

	Object Descriptor	Object Identifier	D	I	R
1m	FTAM PCI	{ISO standard 8571 abstact-syntax (2) ftam-pci (1)}	m	Y	Y
2	FTAM FADU	{ISO standard 8571 abstact-syntax (2) ftam-fadu (2)}	o	/	/
3m		{joint-ISO-ccitt association-control (2) abstract-syntax(1) apdus(0) version1 (1)}	m	Y	Y
4m	FTAM unstructured text abstract syntax	{ISO standard 8571 abstract-syntax(2) unstructured-text(3)}	m	Y	Y
5m	FTAM unstructured binary abstract syntax	{ISO standard 8571 abstract-syntax(2) unstructured-binary(4)}	m	Y	Y
6	NBS file directory entry abstract syntax	{ISO identified organisation icd(9999)organisation-code(1) abstact-syntax(2)nbs-ass(2)	—	Y	Y
7	INTAP abstract syntax AS1	ISO member-body jisc(392)ftam(10)	—	/	/

Note that ISO 8571 requires the presence of the transfer syntax derived from the ‘‘Basic Encoding of a single ASN.1 type’’ {joint-ISO-ccitt basic-encoding (1)} encoding rules for the transfer of the ‘‘FTAM PCI’’ and the ‘‘FTAM FADU’’ abstract syntaxes. Implementation detail of this transfer syntax, and other transfer syntaxes supported, is specified in the PICS of ISO 8823.

8.2.4 Section Four: Virtual Filestore Detail

A.10 Virtual Filestore

This clause details the conformance to the file model, file attribute support and to file structure support. State whether the hierarchical file model (see ISO 8571-2) is supported, and if so, which constraint sets and, where relevant, the maximum depth of hierarchy supported.

A.10.1 File Model

	FILE MODEL	D	R	
1m	Hierarchical	o	Y	
2	Other models (specify or detail in an appendix)	—	N	

A.10.2 Attribute

A.10.2.1 Attribute Groups Implemented

State which file attribute groups are implemented, and which are not implemented. The level of support within each group is to be stated in **Clause A.10.2.2**.

	ATTRIBUTE GROUP NAME	D	I	R	
1m	Kernel	m	Y	Y	
2	Storage	o	Y	Y	
3	Security	o	/	/	
4	Private	o	/	/	

A.10.2.2 Attribute Values

Complete the tables for all attribute groups, shown as supported in **clause A.10.2.1**, indicating for the initiator role whether the attribute is fully or partially supported. If a group is implemented the range of values of each attribute in the group shall be stated in the ‘‘RANGE OF VALUES’’ column, or a forward reference included, possibly to an appendix, giving further details of the supported value range.

Conformance to ISO 8571 requires, that for attribute groups supported, at least the minimum range of attribute values defined in ISO 8571-2, be supported.

An initiator shall not partially support attributes.

On any single line in a responder table in **Clause A.10.2** an entry shall only be made for R (full) or R (partial) not for both.

	KERNEL GROUP (INITIATOR)	D	I full	RANGE OF VALUES
1m	Filename	f	Y	see **A.10.2.3**
2m	Permitted Actions	f	Y	
3m	Contents Type	f	Y	see **A.12.7**

	KERNEL GROUP (RESPONDER)	D	R full	RANGE OF VALUES
4m	Filename	f	Y	see **A.10.2.3**
5m	Permitted Actions	f	Y	
6m	Contents Type	f	Y	see **A.12.7**

	STORAGE GROUP (INITIATOR)	D	I full	RANGE OF VALUES
7m	Storage Account	f	Y	
8m	File availability	f	Y	
9m	Future Filesize	f	Y	

	STORAGE GROUP (RESPONDER)	D	R full	R partial
10	Storage Account	p	/	Y
11	Date and time of creation	p	/	Y
12	Date and time of last modification	p	Y	N
13	Date and time of last read access	p	Y	N
14	Date and time of last attribute modification	p	Y	N
15	Identity of creator	p	Y	N
16	Identity of last modifier	p	/	Y
17	Identity of last reader	p	/	Y
18	Identity of last attribute modifier	p	/	Y
19m	File availability	p	/	Y
20m	Filesize	p	Y	N
21	Future Filesize	p	/	Y

	SECURITY GROUP (INITIATOR)	D	I full	RANGE OF VALUES
22m	Access Control	f	/	see **A.12.2**
23m	Legal Qualifications	f	/	

	SECURITY GROUP (RESPONDER)	D	R full	R partial	RANGE OF VALUES
24m	Access Control	p	/	/	see **A.12.2**
25	Legal Qualifications	p	/	/	

	PRIVATE GROUP (INITIATOR)	D	I full	RANGE OF VALUES
35	Private Use	f	/	

Note that if the private attribute is implemented, then details of the values and their semantics shall be stated in an appendix.

	PRIVATE GROUP (RESPONDER)	D	R full	R partial	RANGE OF VALUES
36	Private Use	p	/	/	

Note that if the private attribute is implemented, then details of the values and their semantics shall be stated in an appendix.

A.10.2.3 Filename Detail

Specify what restrictions, if any, apply to the filename.

	FILENAME - INITIATOR	
1	How many filename elements are supported	1
2	Which G sets of which character sets are supported	ISO 646 IRV G0
3	Which characters, if any, are excluded from the character set	
4	Which is the maximum string length of each filename element	_POSIX_PATH_MAX (255 bytes - see note)
5	Other restrictions	

Note that this value is defined in the referenced **POSIX Standard** *(see* **Referenced Documents***). The question of whether this value includes or excludes the null character used to terminate strings is the subject of a currently un-published interpretation by the IEEE Vice-Chair on Interpretations. As a result of this interpretation, the value and its description is likely to change in a future technical revision to the referenced* **POSIX Standard** *(see Referenced Documents).*

Specify what restrictions, if any, apply to the filename.

	FILENAME - RESPONDER	
6	How many filename elements are supported	1
7	Which G sets of which character sets are supported	implementation_specific
8	Which characters, if any, are excluded from the character set	—
9	Which is the maximum string length of each filename element	implementation_specific
10	Other restrictions	

A.10.3 File Structures

A.10.3.1 Constraint Sets

If the Contents type ‘‘Abstract Syntax / Constraint Set’’ is implemented, state which constraint sets are, and which are not, implemented. Where applicable an integer figure shall be supplied to indicate the maximum depth of hierarchy which the implementation supports.

	CONSTRAINT SET NAME	D	I	R	DEPTH
1m	Unstructured	o	Y	Y	NOT APPLICABLE
2	Sequential Flat	o	/	/	NOT APPLICABLE
3	Ordered flat	o	/	/	NOT APPLICABLE
4	Ordered flat with unique names	o	/	/	NOT APPLICABLE
5	Ordered hierarchical	o	/	/	
6	General hierarchical	o	/	/	
7	General hierarchical with unique names	o	/	/	

Note that if the action allowed and/or the permissible FADU identities are restricted within a specific constraint set beyond the restrictions given by the specified support of the permitted actions file attribute and the document type supported, such restrictions shall be stated in an appendix.

A.10.3.2 File and Filestore Actions

A.10.3.2.1 Filestore Actions

Support for filestore actions is depended upon the functional units implemented (see **Clause A.12.4** and **Clause A.12.5**).

A.10.3.2.2 File Actions

State whether the action is, or is not, supported within a file open regime, for the constraint sets implemented in the responder role.

		CONSTRAINT SET		
		un- structured		
	ACTION	D	R	
1	Locate	—	—	
2	Read	o	Y	see note
3	Insert	—	—	
4	Replace	o	Y	see note
5	Extend	o	Y	see note
6	Erase	o	/	

Note that the support of at least one of read, replace or extend is required.

A.10.3.2.3 Access Contexts Implemented

	ACTION CONTEXT	un-structured	
		D	R
1	US	—	—
2m	UA	o	Y
3	FS	—	—
4	FL	—	—
5	FA	—	—
6	HN	—	—
7	HA	—	—

A.10.4 Additional Information

State whether there are any circumstances under which the existence of a file, its contents, or the values of the supported attributes may change, between separate accesses using the FTAM protocol.

1 State details here or in an appendix

State whether there are any circumstances under which modifications to the file contents or the values of the file attributes by FTAM protocol exchanges may not subsequently be available for use.

1 State details here or in an appendix

A.10.5 Override

State the implemented filestore capability.

	Responder Override	D	R
1m	Create failure	o	Y
2m	Select old file	o	Y
3	Delete and create with old attribute	o	/
4m	Delete and create with new attribute	o	Y

Note that the specification of the role of initiator is to be given in section five (file protocol detail).

8.2.5 Section Five: File Protocol Details

A.11 File Protocol

Detail the level of support for the FTAM protocol and its PDU fields. State which fields are, and which are not, implemented in each PDU.

If a PDU field is implemented, its range of values shall be specified, or, if applicable, the reference completed. Fields not implemented shall be so marked.

Clauses A.11.2 to **A.11.24** require an indication of which PDUs are implemented. The conformance requirements for PDUs are dependent on the particular functional units implemented. PDUs indicated in **Clauses A.11.8** to **A.11.24** as conditional shall be considered mandatory when a particular functional unit is implemented, according to the following table.

PDUs	Clauses	Functional Units								
		Kernel	Read	Write	Access	L.F.M	E.F.M.	Grouping	Recovery	Restart
F-CREATE	**A.11.8**	/	/	/	/	Y	/	/	/	/
F-DELETE	**A.11.9**	/	/	/	/	Y	/	/	/	/
F-READ-ATTRIB	**A.11.10**	/	/	/	/	Y	/	/	/	/
F-CHANGE-ATTRIB	**A.11.11**	/	/	/	/	/	Y	/	/	/
F-OPEN	**A.11.12**	/	Y	Y	/	/	/	/	/	/
F-CLOSE	**A.11.13**	/	Y	Y	/	/	/	/	/	/
F-BEGIN-GROUP	**A.11.14**	/	/	/	/	/	/	Y	/	/
F-END-GROUP	**A.11.15**	/	/	/	/	/	/	Y	/	/
F-RECOVER	**A.11.16**	/	/	/	/	/	/	/	Y	/
F-LOCATE	**A.11.17**	/	/	/	Y	/	/	/	/	/
F-ERASE	**A.11.18**	/	/	/	Y	/	/	/	/	/
F-READ	**A.11.19**	/	Y	/	/	/	/	/	/	/
F-WRITE	**A.11.20**	/	/	Y	/	/	/	/	/	/
F-DATA-END	**A.11.21**	/	Y	Y	/	/	/	/	/	/
F-TRANSFER-END	**A.11.22**	/	Y	Y	/	/	/	/	/	/
F-CANCEL	**A.11.23**	/	Y	Y	/	/	/	/	/	/
F-RESTART	**A.11.24**	/	/	/	/	/	/	/	/	Y

NOTES

1. *In order to keep the protocol tables compact some forward references have been introduced to the clauses which expand upon the detail of file support.*
2. *The FTAM protocol will require a number of optional layer services to be available (e.g. Application Entity Title in ACSE). This requirement is outside the scope of this PICS pro forma.*

A.11.1 GraphicString Support

State the supported G sets in the **GraphicString** of the PDU fields implemented and the maximum length of the string, if such a restriction applies. If the support for initiator and responder roles are not the same state each restriction separately.

1	GO

NOTES

1. *If the level of support of this feature varies with the different fields in which the GraphicString is used full details shall be given in an appendix.*
2. *The character set supported for Document Types is specified in* **ISP 10607-3**, **Section 6**, **Document Types**.

A.11.2 FTAM Regime Establishment

	F-INITIALISE PDU	D	I	D	R	
1m		m	Y	m	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2m	State result	—	—	m	Y	all values defined in ISO 8571
3m	Action result	—	—	m	Y	all values defined in ISO 8571
4m	Protocol version	m	Y	m	Y	see **Section 2**
5	Implementation information	o	/	o	/	see **A.12.1**
6m	Presentation context management	m	Y	m	Y	see note 1
7m	Service class	m	Y	m	Y	see **A.12.4**
8m	Functional units	m	Y	m	Y	see **A.12.5**
9m	Attribute groups	m	Y	m	Y	see **A.10.2**
10	Shared ASE information	o	/	o	/	see **A.12.9**
11m	FTAM Quality of Service	m	Y	m	Y	see **A.12.8**
12m	Contents type list	o	Y	o	Y	see **A.12.7.1**
13m	Initiator identity	o	Y	—	—	
14	Account	o	Y	—	—	
15m	Filestore password	o	Y	—	—	see **A.12.11**
16m	Diagnostic	—	—	o	Y	see **A.12.6**
17m	Checkpoint window	m	Y	m	Y	see note 2

NOTES

1. *The values available for the presentation context management field depend upon the functional units implemented in ISO 8823.*
2. *Checkpoint window field is indicated as mandatory in accordance with ISO 8571-4. The field is defaulted to the value 1.*

A.11.3 FTAM Regime Termination (Orderly)

	F-TERMINATE PDU	D	I	D	R	
1m		m	Y	m	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2	Shared ASE information	o	/	o	/	see **A.12.9**
3	Charging	—	—	o	/	see **A.12.10**

A.11.4 FTAM Regime Termination (Abrupt) by Service User

	F-U-ABORT PDU	D	I	R	
1m		m	Y	Y	
	FIELD NAME				RANGE OF VALUES OR REFERENCE
2m	Action result	m	Y	Y	all values defined in ISO 8571
3m	Diagnostic	o	Y	Y	see **A.12.6**

A.11.5 FTAM Regime Termination (Abrupt) by Service Provider

	F-U-ABORT PDU	D	I	R	
1m		m	Y	Y	
	FIELD NAME				RANGE OF VALUES OR REFERENCE
2m	Action result	m	Y	Y	all values defined in ISO 8571
3m	Diagnostic	o	Y	Y	see **A.12.6**

A.11.6 File Selection

	F-SELECT PDU	D	I	D	R	
1m		m	Y	m	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2m	State result	—	—	m	Y	all values defined in ISO 8571
3m	Action result	—	—	m	Y	all values defined in ISO 8571
4m	Attributes	m	Y	m	Y	see **A.10.2**
5m	Requested access	m	Y	—	—	see **A.12.16**
6m	Access password	o	N	—	—	see **A.12.12**
7	Concurrency control	o	/	—	—	see **A.12.13**
8	Shared ASE information	o	/	o	/	see **A.12.9**
9	Account	o	/	—	—	
10m	Diagnostic	—	—	o	Y	see **A.12.6**

A.11.7 File Deselection

	F-DESELECT PDU	D	I	D	R	
1m		m	Y	m	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2m	Action result	—	—	m	Y	all values defined in ISO 8571
3	Charging	—	—	o	/	see **A.12.10**
4	Shared ASE information	o	/	o	/	see **A.12.9**
5m	Diagnostic	—	—	o	Y	see **A.12.6**

A.11.8 File Creation

	F-CREATE PDU	D	I	D	R	
1		o	Y	o	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2m	State result	—	—	m	Y	all values defined in ISO 8571
3m	Action result	—	—	m	Y	all values defined in ISO 8571
4m	Override	m	Y	—	—	see **A.12.15**
5m	Initial attributes	m	Y	m	Y	see **A.10.2**
6m	Create password	o	Y	—	—	see **A.12.12**
7m	Requested access	m	Y	—	—	see **A.12.16**
8m	Access passwords	o	/	—	—	see **A.12.3.5**
9	Concurrency control	o	/	—	—	see **A.12.13**
10	Shared ASE information	o	/	o	/	see **A.12.9**
11	Account	o	/	—	—	
12m	Diagnostic	—	—	o	Y	see **A.12.6**

A.11.9 File Deletion

	F-DELETE PDU	D	I	D	R	
1m		o	Y	o	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2m	Action result	—	—	m	Y	all values defined in ISO 8571
3	Shared ASE information	o	/	o	/	see **A.12.9**
4	Charging	—	—	o	/	see **A.12.10**
5m	Diagnostic	—	—	o	Y	see **A.12.6**

A.11.10 Read Attributes

	F-READ-ATTRIB PDU	D	I	D	R	
1m		o	Y	o	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2m	Action result	—	—	m	Y	all values defined in ISO 8571
3m	Attribute names	m	Y	—	—	
4m	Attributes	—	—	o	Y	see **A.10.2**
5m	Diagnostic	—	—	o	Y	see **A.12.6**

A.11.11 Change Attributes

	F-CHANGE-ATTRIB PDU	D	I	D	R	
1m		o	Y	o	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2m	Action result	—	—	m	Y	all values defined in ISO 8571
3m	Attributes	m	Y	o	Y	see **A.10.2**
4m	Diagnostic	—	—	o	Y	see **A.12.6**

A.11.12 File Open

	F-OPEN PDU	D	I	D	R	
1m		o	Y	o	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2m	State result	—	—	m	Y	all values defined in ISO 8571
3m	Action result	—	—	m	Y	all values defined in ISO 8571
4m	Processing mode	m	Y	—	—	see **A.12.17**
5m	Contents type	m	Y	m	Y	see **A.12.7**
6	Concurrency control	o	/	o	/	see **A.12.13**
7	Shared ASE information	o	/	o	/	see **A.12.9**
8m	Enable FADU locking	m	Y	—	—	
9	Activity identifier	o	/	—	—	
10m	Diagnostic	—	—	o	Y	see **A.12.6**
11m	Recovery mode	m	Y	m	Y	see **A.12.18**
12	Remove contexts	o	/	—	—	max number of presentation contexts =
13	Define contexts	o	/	—	—	
14m	Presentation action	—	—	m	Y	see notes

NOTES

1. *The values available for the presentation action field depend upon the functional units implemented in ISO 8823.*
2. *Presentation action field is indicated as mandatory in accordance with ISO 8571#4. The field is defaulted to no action.*
3. *Enable FADU Locking field is indicated as mandatory in accordance with ISO 8571-4. The field is defaulted to false.*

A.11.13 File Close

	F-CLOSE PDU	D	I	D	R	
1m		o	Y	o	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2m	Action result	m	Y	m	Y	all values defined in ISO 8571
3	Shared ASE information	o	/	o	/	see **A.12.9**
4m	Diagnostic	o	Y	o	Y	see **A.12.6**

A.11.14 Beginning of Grouping

	F-BEGIN-GROUP PDU	D	I	D	R	
1m		o	Y	o	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2m	Threshold	m	Y	m	Y	

A.11.15 End of Grouping

	F-END-GROUP PDU	D	I	D	R	
1m		o	Y	o	Y	
	The F-END-GROUP PDU carries no fields					

A.11.16 Regime Recovery

/

A.11.17 Locate File Access Data Unit

/

A.11.18 Erase File Access Data Unit

/

A.11.19 Read Bulk Data

1	F-READ PDU	D o	I Y	
	FIELD NAME			RANGE OF VALUES OR REFERENCE
2m	FADU identity	m	Y	
3m	Access context	m	Y	see **A.10.3.2.3**
4	FADU lock	o	/	see **A.12.14**

A.11.20 Write Bulk Data

	F-WRITE PDU	D	I	
1		o	Y	
	FIELD NAME			RANGE OF VALUES OR REFERENCE
2m	FADU operation	m	Y	
3m	FADU identity	m	Y	
4	FADU lock	o	/	see **A.12.14**

A.11.21 End of Data Transfer

	F-DATA-END PDU	D	I	R	
1		o	Y	Y	
	FIELD NAME				RANGE OF VALUES OR REFERENCE
2m	Action result	m	Y	Y	all values defined in ISO 8571
3m	Diagnostic	o	Y	Y	see **A.12.6**

A.11.22 End of Transfer

	F-TRANSFER-END PDU	D	I	D	R	
1m		o	Y	o	Y	
	FIELD NAME					RANGE OF VALUES OR REFERENCE
2m	Action result	—	—	m	Y	all values defined in ISO 8571
3	Shared ASE information	o	/	o	/	see **A.12.9**
4m	Diagnostic	—	—	o	Y	see **A.12.6**

A.11.23 Cancel Data Transfer

	F-CANCEL PDU	D	I	R	
1m		o	Y	Y	
	FIELD NAME				RANGE OF VALUES OR REFERENCE
2m	Action result	m	Y	Y	all values defined in ISO 8571
3	Shared ASE information	o	/	/	see **A.12.9**
4m	Diagnostic	o	Y	Y	see **A.12.6**

A.11.23.1 F-CANCEL Mapping

If the following conditions are met ISO 8571 requires that the F-CANCEL maps to P-RESYNC

a. either or both the read and write functional units are implemented

b. P-RESYNC and P-SYNC-MINOR functional units are available. See ISO 8023 PICS.

1	The F-CANCEL service primitive maps to P-RESYNC YES / NO	/

A.11.24 Restart Data Transfer

/

A.12 Expanded PDU Field Detail

This clause identifies further PDU field and filestore detail to expand on that given in 10 and 11.

A.12.1 Implementation Information Detail

Complete the following if support is claimed for the implementation information field of the F-INITIALISE request or response PDUs.

	Initiator
1	State the G sets implemented and the maximum length of string
2	Optional details of the semantics of this parameter (may be included in an appendix)

Note that this parameter is not subject to conformance testing.

A.12.2 Access Control Detail

/

A.12.3 Access Control Element Detail

A.12.3.1 Action list detail (initiator)

Void.

A.12.3.2 Action list detail (responder)

Void.

A.12.3.3 Concurrency access term

/

A.12.3.4 Identity term

Void.

A.12.3.5 Initiator Access Passwords

/

A.12.3.6 Responder Access Passwords

/

A.12.3.7 Location term

Void.

A.12.3.7.1 Application Entity Titles detail

Void.

A.12.3.8 Access control element combinations

/

A.12.4 Service Class Field Detail

		D	I	R
1m	Transfer class	o	Y	Y
2	Access class	o	/	/
3m	Management class	o	/	Y
4m	Transfer and management class	o	Y	Y
5	Unconstrained class	o	/	/

Note that the initiator is only permitted to specify those combinations defined in ISO 8571-3.

A.12.5 Functional Unit Field Detail

State the functional units implemented in each service class.

	FUNCTIONAL UNITS	SERVICE CLASSES								
		Transfer			Management			Transfer and Management		
		D	I	R	D	I	R	D	I	R
1m	Kernel	m	Y	Y	m	/	Y	m	Y	Y
2	Read (see note 2)	c	Y	Y	—	—	—	c	Y	Y
3	Write (see note 2)	c	Y	Y	—	—	—	c	Y	Y
4	File access	—	—	—	—	—	—	—	—	—
5m	Limited File Management	o	Y	Y	m	/	Y	m	Y	Y
6m	Enhanced File Management	o	Y	Y	o	/	Y	o	Y	Y
7m	Grouping	m	Y	Y	m	/	Y	m	Y	Y
8	FADU Locking	—	—	—	—	—	—	—	—	—
9	Recovery	o	/	/	o	/	/	o	/	/
10	Restart	o	/	/	o	/	/	o	/	/

NOTES

1. *the recovery and the restart functional units are only available at the internal file service interface and should only be explicitly referenced in the protocol.*
2. *the C indicates that either or both of the read and write functional units shall be implemented in the particular service class.*

A.12.6 Diagnostic Field Detail

This clause shall be completed if any diagnostic field is indicated as implemented. The following tables assume that all diagnostic fields have the same implementation details. If this is not so the tables of this clause shall be repeated as necessary in an appendix.

		D	I	R
1m	Diagnostic type	m	Y	Y
2m	Error identifier	m	Y	Y
3m	Error observer	m	Y	Y
4m	Error source	m	Y	Y
5	Suggested delay	o	/	/
6m	Further details	o	Y	Y
7	If the further details parameter is implemented, detail the character sets and maximum string length, if applicable.			

A.12.7 Contents Type Detail

This clause shall be completed if any of the following are implemented:

a. the contents type list field of F-INITIALISE

b. the contents type field is implemented in F-OPEN

c. the contents type field is implemented in F-RECOVER

A.12.7.1 Contents Type List Parameter

If the implementation supports the content type list field on the F-INITIALISE PDU complete the following support details:

		D	I	R	Maximum number of elements
1m	document type specification	o	Y	Y	3
2m	abstract syntax specifications	o	Y	Y	3

A.12.7.2 Contents Type Parameter

If the implementation supports the contents type field on the F-OPEN, F-CREATE and/or F-RECOVER PDUs complete the following support details:

		D	I	R
1m	document type specifications	o	Y	Y
2	abstract syntax/constraint set pair specifications	o	/	/

Note that the detail of document types supported is contained in **ISP 10607-3**, **Section 6**, **Document Types**.

A.12.8 FTAM Quality of Service Details

A.12.8.1 Initiator

State all possible combinations of FTAM quality of service and recovery or restart functional units which the initiator is capable of generating.

	Initiator	Functional units (in request)							
	FTAM Quality of Service	Neither		Restart		Recovery		Both	
1m	No recovery	m	Y	—	—	—	—	—	—
2	Class 1	o	/	o	/	o	/	o	/
3	Class 2	o	/	o	/	o	/	o	/
4	Class 3	o	/	o	/	o	/	o	/

A.12.8.2 Responder

State what responses may be returned by the responder implementation for all valid incoming request combinations of FTAM quality of service and the recovery or restart functional units.

	Request		Response										
	FTAM QoS supported Values	Recovery or restart FUs Supported	FTAM QoS values	supported FTAM QoS values		Functional Units							
						Neither		Restart		Recovery		Both	
1	0	Neither	0	m	Y	m	Y	—	—	—	—	—	—
2	1	Neither	1	o	/	o	/	—	—	—	—	—	—
3	1	Neither	0	o	Y	o	Y	—	—	—	—	—	—
4	1	Restart	1	o	/	o	/	o	/	—	—	—	—
5	1	Restart	0	o	/	o	/	o	/	—	—	—	—
6	1	Recovery	1	o	/	o	/	—	—	o	/	—	—
7	1	Recovery	0	o	Y	o	/	—	—	o	/	—	—
8	1	Both	1	o	/	o	/	o	/	o	/	o	/
9	1	Both	0	o	/	o	/	o	/	o	/	o	/
10	2	Neither	2	o	/	o	/	—	—	—	—	—	—
11	2	Neither	1	o	/	o		—	—	—	—	—	—
12	2	Neither	0	o	Y	o	Y	—	—	—	—	—	—
13	2	Restart	2	o	/	o	/	o	/	—	—	—	—
14	2	Restart	1	o	/	o	/	o	/	—	—	—	—
15	2	Restart	0	o	/	o	/	o	/	—	—	—	—
16	2	Recovery	2	o	/	o	/	—	—	o	/	—	—
17	2	Recovery	1	o	/	o	/	—	—	o	/	—	—
18	2	Recovery	0	o	/	o	/	—	—	o	/	—	—

	Request		Response										
	FTAM QoS supported Values	Recovery or restart FUs Supported	FTAM QoS values	supported FTAM QoS values		Functional Units							
						Neither		Restart		Recovery		Both	
19	2	Both	2	o	/	o	/	o	/	o	/	o	/
20	2	Both	1	o	/	o	/	o	/	o	/	o	/
21	2	Both	0	o	/	o	/	o	/	o	/	o	/
22	3	Neither	3	o	/	o	/	—	—	—	—	—	—
23	3	Neither	2	o	/	o	/	—	—	—	—	—	—
24	3	Neither	1	o	/	o	/	—	—	—	—	—	—
25	3	Neither	0	o	Y	o	Y	—	—	—	—	—	—
26	3	Restart	3	o	/	o	/	o	/	—	—	—	—
27	3	Restart	2	o	/	o	/	o	/	—	—	—	—
28	3	Restart	1	o	/	o	/	o	/	—	—	—	—
29	3	Restart	0	o	/	o	/	o	/	—	—	—	—
30	3	Recovery	3	o	/	o	/	—	—	o	/	—	—
31	3	Recovery	2	o	/	o	/	—	—	o	/	—	—
32	3	Recovery	1	o	/	o	/	—	—	o	/	—	—
33	3	Recovery	0	o	/	o	/	—	—	o	/	—	—
34	3	Recovery	3	o	/	o	/	o	/	o	/	o	/
35	3	Recovery	2	o	/	o	/	o	/	o	/	o	/
36	3	Recovery	1	o	/	o	/	o	/	o	/	o	/
37	3	Recovery	0	o	/	o	/	o	/	o	/	o	/

Alternatively details of the response FTAM QoS details may be attached as an appendix.

A.12.9 Details of Shared ASE Information

The mode of use of the shared ASE information field will be highly dependent upon the nature of the symbiotic ASE. Implementations claiming support of the shared ASE information field shall include a reference here to an appendix providing complete details of its use.

A.12.10 Details of Charging

State charging details if the charging parameter is implemented.

	Charging	D	R
1m	Resource identifier term	m	/
2m	Charging unit term	m	/
3m	Charging value term	m	/
4	Further charging details including the number of charging elements supported		

A.12.11 Filestore Password Detail

	Filestore password detail	D	I	R
1	OctetString	o	/	/
2	GraphicString	o	Y	/
3	State general initiator restrictions which may apply to the range of filestore passwords (e.g. GraphicString character sets and string lengths).			

A.12.12 Create Password Detail

State what restrictions apply to the range of create passwords supported.

	Create password detail	D	I	R
1	OctetString	o	/	/
2	GraphicString	o	Y	/
3	State general initiator restrictions which may apply to the range of create passwords (e.g. GraphicString character sets and string lengths).			

A.12.13 Concurrency Control

If an implementation does not support access control but does claim support of concurrency control this clause shall be completed to indicate which concurrency locks may be set from the initiator role and which set in the responder role. Further, state the default values applied by the responder, if the concurrency control field is not supported.

A.12.13.1 Implemented Values

	Action	Concurrency control implemented values											
		not required			shared			exclusive			no access		
		D	I	R	D	I	R	D	I	R	D	I	R
1	Read	o	/	/	o	Y	Y	o	/	/	o	/	/
2	Insert	o	/	/	o	/	/	o	/	/	o	/	/
3	Replace	o	/	/	o	/	/	o	Y	Y	o	/	/
4	Extend	o	/	/	o	/	/	o	Y	Y	o	Y	Y
5	Erase	o	/	/	o	/	/	o	/	/	o	/	/
6	Read attrib	o	/	/	o	Y	Y	o	/	/	o	/	/
7	Change attrib	o	/	/	o	/	/	o	Y	Y	o	/	/
8	Delete file	o	/	/	o	/	/	o	Y	Y	o	/	/

A.12.13.2 Responder Default Values

	Action	Concurrency control responder default values							
		not required		shared		exclusive		no access	
		D	R	D	R	D	R	D	R
1	Read	o	/	o	Y	o	/	o	/
2	Insert	o	/	o	/	o	/	o	/
3	Replace	o	/	o	/	o	Y	o	/
4	Extend	o	/	o	/	o	Y	o	/
5	Erase	o	/	o	/	o	/	o	/
6	Read attrib	o	Y	o	Y	o	/	o	/
7	Change attrib	o	/	o	/	o	Y	o	/
8	Delete file	o	/	o	/	o	Y	o	/

A.12.14 FADU Locking

If FADU locking is implemented, complete the following table to indicate which FADU concurrency locks may be set from the implementation while FADU locking is enabled from the F-OPEN assuming no additional access control restrictions.

		FADU Locking Support Values											
		not required			shared			exclusive			no access		
		D	I	R	D	I	R	D	I	R	D	I	R
1	Read	o	/	/	o	/	/	o	/	/	o	/	/
2	Insert	o	/	/	o	/	/	o	/	/	o	/	/
3	Replace	o	/	/	o	/	/	o	/	/	o	/	/
4	Extend	o	/	/	o	/	/	o	/	/	o	/	/
5	Erase	o	/	/	o	/	/	o	/	/	o	/	/

A.12.15 Initiator Override

State which values of override are implemented in the role of initiator.

	Initiator override	D	I
1	Create failure	o	Y
2	Select old file	o	Y
3	Delete and recreate with old attributes	o	/
4	Delete and create with new attributes	o	Y

Note that the specification of the role of a responder is to be given in **Clause A.10.5**, *the filestore section.*

A.12.16 Requested Access

	Action	D	I	R
1	Read	o	Y	Y
2	Insert	o	/	/
3	Replace	o	Y	Y
4	Extend	o	Y	Y
5	Erase	o	N	N
6m	Read attribute	o	Y	Y
7m	Change attribute	o	Y	Y
8m	Delete file	o	Y	Y

A.12.17 Processing Mode

State which processing modes are capable of being specified, and which accepted. The validity and applicability of these for any instances of communication will be subject at least to the constraints of ISO 8571.

	Processing mode	D	I	R	
1	Read	o	Y	Y	
2	Insert	o	/	/	
3	Replace	o	Y	Y	
4	Extend	o	Y	Y	
5	Erase	o	N	N	

A.12.18 Recovery Mode

If the recovery mode parameter is implemented the implementation value for initiator and responder shall be stated here.

	Recovery mode	D	I	R
1	None	o	Y	Y
2	At start of file	o	/	/
3	Any active checkpoint	o	/	/

8.2.6 Section Six: Document Types

A.13 Document Types

Conformance to document types is given at two levels. The following tables shall be completed to indicate which document types have some level of support. The detail of that level of support shall be stated in the following sections.

	Entry number	FTAM-1	D	I	R
1m	object descriptor	ISO FTAM unstructured text	o	Y	Y
	object identifier	{ISO standard 8571 document-type-(5) unstructured-text (1)}			

	Entry number	FTAM-2	D	I	R
2	object descriptor	ISO FTAM sequential text	o	/	/
	object identifier	{ISO standard 8571 document-type-(5) sequential-text (2)}			

	Entry number	FTAM-3	D	I	R
3m	object descriptor	ISO FTAM unstructured binary	o	Y	Y
	object identifier	{ISO standard 8571 document-type-(5) unstructured-binary (3)}			

	Entry number	FTAM-4	D	I	R
4	object descriptor	ISO FTAM sequential binary	o	/	/
	object identifier	{ISO standard 8571 document-type-(5) sequential-binary (4)}			

In the following tables state the detailed level of support for each document type for which a level of support has been indicated above.

Registration information, when available, shall be included.

NOTES

1. *The implementation of additional document types may be specified in a similar fashion. Registration information, when available, shall be included.*
2. *No entry is specified for* FTAM-5 *because* FTAM-5 *is an incomplete document type, and therefore cannot be implemented on its own.*

The following are additional document types defined in ISP 10607-3.

	Entry number	NBS-9	D	I	R
5	object descriptor object identifier	NBS-9 FTAM file directory file {ISO identified-organisation id (9999) organisation-code(1) document-type(5) file-directory(9)}	—	Y	Y

	Entry number	INTAP-1	D	I	R
6	object descriptor object identifier	INTAP record file {ISO member-body jisc (392) ftam(10) document-type(2) intap-record file(5)}	—	/	/

A.13.1 FTAM-1

A.13.1.1 Universal Class Number Parameter

			D	I	R	Reference
1m	Universal class number parameter supported		o	Y	Y	
2	PrintableString	Universal class 19	o	/	/	
3	TeletexString	Universal class 20	o	/	/	see **A.13.1.5**
4	VideoString	Universal class 21	o	/	/	see **A.13.1.5**
5m	IA5String	Universal class 22	o	Y	Y	
6m	GraphicString	Universal class 25	o	Y	Y	see **A.13.1.4**
7m	VisibleString	Universal class 26	o	Y	Y	
8m	GeneralString	Universal class 27	o	Y	Y	see **A.13.1.5**

A.13.1.2 String Length Parameter and String Significance Parameter Combinations

		D	I	R	Maximum string length parameter value supported
1m	Maximum string length parameter supported and variable length strings supported	o	Y	Y	
2m	Maximum string length parameter supported and fixed length strings supported	o	Y	Y	
3m	Maximum string length parameter supported and not significant strings supported	o	Y	Y	
4m	Maximum string length parameter NOT supported and not significant strings supported	o	Y	Y	not applicable
5m	Maximum string length parameter NOT supported and variable length strings supported	o	Y	Y	not applicable

A.13.1.3 G Sets Supported

State which G sets are supported in the FTAM-1 GraphicString.

1m	1) ISO 646 IRV (character set registration number 002 in G0) 2) ISO 8859/1 (character set registration numbers 006, 100 in G0, G1 respectively) 3) Other character sets by regional or bi-party agreement

A.13.1.4 G and C Sets Supported

State which G and which C sets are supported in the FTAM-1.

1m	The Graphic sets ISO 646 IRV and ISO 8859/1, supported as described in **A.13.1.3** may be used in conjunction with the control set ISO 646 control characters (character set registration number 001 in C0). When a single format effector is allowed to effect both a single (vertical or horizontal) and a combined vertical and horizontal movement, implementations shall not use the single format effector in the transferred data for effecting the combined vertical and horizontal movement.

A.13.2 FTAM-2

Void.

A.13.3 FTAM-3

A.13.3.1 String Length Parameter and String Significance Parameter Combinations

		D	I	R	Maximum string length parameter value supported
1	Maximum string length parameter and variable length strings	o	/	/	
2	Maximum string length parameter and fixed length strings	o	/	/	
3m	Maximum string length parameter and not significant length strings	o	Y	Y	
4m	Unbounded strings and variable length strings	o	/	/	not applicable
5m	Unbounded strings and not significant length strings	o	Y	Y	not applicable

A.13.4 FTAM-4

Void.

Chapter 9

Lower Upper Layers

9.1 INTRODUCTION

This section defines the requirements of other upper layers of the FTAM stack, i.e., ACSE, Presentation Layer and Session Layer. ISO 8571-5 does not contain PICSs for these layers, but generic PICS (i.e., PICS that are non FTAM specific) are defined in the appropriate ISO standards. ISP 10607-1 includes these PICS, partly filled in for FTAM.

9.2 ACSE - Association Control Service Element

Support for ACSE conforms to the specification of ISP 10607-1. A PICS is not included in this document as there are no optional ACSE protocol elements / parameters for FTAM.

9.3 PRESENTATION LAYER

Support for Presentation conforms to the specification of ISP 10607-1. Options at this layer are determined by the options selected at the FTAM layer. A PICS is not included in this report as there are only minor optional Presentation Protocol elements / parameters for FTAM which are summarised below:

- Support for NBS-AS2 abstract syntax (needed for NBS-9 document types). This is supported.
- Support for INTAP abstract syntax AS1 (needed for INTAP-1 document types). This is not supported.
- Support of INTAP transfer syntax TS1 (needed for INTAP-1 document types). This is not supported.
- Support for RESYNCHRONISATION Functional Unit. This is needed to allow mapping of the F-CANCEL. This is outside the scope of BSFT.
- Support of presentation selector(s) in:

  — Connect Presentation

  — Connect Presentation Accept

  — Connect Presentation Reject PPDUs.

  The calling-presentation-selector is optional for Initiators while the called-presentation-selector and responding presentation-selector are optional for Responders.

All of the above are supported in BSFT. The presentation selector has the ASN.1 type OCTETSTRING. It is generated by BSFT Initiators and processed by BSFT Responders as described earlier. When generated by BSFT, the presentation selector must be limited to 4 bytes for compatibility with ISP 10607-3.

9.4 SESSION LAYER

Support for Session conforms to the specification of ISP 10607-1. Options at this layer are determined by the options selected at the FTAM and Presentation layers. A PICS is not included in this document as there are only minor optional Session Protocol elements/parameters for FTAM. These are summarised below:

- BSFT does not require the support of the Minor Synchronisation and Resynchronisation Functional Units (these are needed in order to map the F-CANCEL onto the P-RESYNCHRONISE).
- BSFT always requests the use of the Transport Expedited Service, but is prepared for it to be refused by the Responder. BSFT Responders will accept the Transport Expedited Service if proposed.
- BSFT does not re-use the transport connection.
- BSFT implementations do not attempt to negotiate the use of session segmenting (and hence do not use the TSDU maximum size parameter).
- The initial serial number and initial token setting item parameters are not used, since the Minor Synchronisation and Resynchronisation Functional Units are not used.
- The transport disconnect parameter in the REFUSE, FINISH and ABORT SPDUs always indicates that the transport connection will not be reused.
- The reflect parameter values in the ABORT SPDU is implemented. This should aid the trace of error conditions.
- The enclosure item in the data SPDU is not supported, since segmenting is not supported.
- BSFT does not require the support of the MINOR SYNC POINT and RESYNCHRONISE SPDUs.
- The GIVE TOKENS SPDU is implemented.

Appendix A

BSFT User Interface Definition

This section defines the user interface for the BSFT facility.

NAME

BSFT - byte stream file transfer program

SYNOPSIS BSFT [-v] [-d] [-i] [-n] [-g] [-t] [host]

DESCRIPTION

BSFT is the user interface for X/Open compliant OSI file transfer systems. BSFT transfers files to and from a remote network site. The two ends of a BSFT connection are asymmetric. The end that supports the user interface is termed the Initiator, while the system with which the Initiator communicates is termed the Responder.

The remote system (the Responder) may be specified on the command line. If this is done, BSFT immediately attempts to establish a connection to a BSFT server on that host; otherwise BSFT enters its command interpreter and awaits instructions from the user. When BSFT is awaiting commands from the user, it displays the prompt *bsft>* .

BSFT is fully conformant with the ISO standards for file transfer, **IS 8571**, **File Transfer Access and Management (FTAM)**. This standard has a rich set of functions and options and in order to encourage interoperability, regional groups (EWOS, NIST and AOW) and ISO have defined standard subsets of FTAM, known as ‘‘profiles’’. BSFT complies with the FTAM profiles: AFT 11 - Simple File Transfer Service (Unstructured) (ISP 10607-3) and AFT 3 - File Management Service profile (ISP 10607-6). These profiles are refered to here as the ‘‘base profiles’’. Conformance to these profiles allows BSFT to interwork with three types of systems:

- X/Open compliant systems
- UNIX systems that do not comply with X/Open standards
- non-UNIX systems.

All three types of systems are handled transparently by BSFT such that the user is generally not aware of the type of system with which they are communicating. However, when communicating with non-X/Open compliant systems it is possible that

- some functions may not be available
- the user may have to specify filenames that are not valid on X/Open systems
- the user may have to specify some parameters that are not required between X/Open systems.

Options

The options may be specified at the command line, or to the command interpreter.

host defines the FTAM Responder with which to connect. The format of this parameter is implementation-defined.

-v show all responses from the remote server, as well as report on data transfer statistics. This is turned on by default if BSFT is running interactively with its input coming from the user’s terminal.

-n do not attempt "auto-login". If auto-login is enabled, BSFT searches the *.ftamrc* file in the user's home directory for an entry describing an account on the remote system. If no entry exists, BSFT prompts for the username of an account on the remote system (the default is the username on the local system). If the password value is missing, the Initiator prompts for a password and does not echo the value input by the user. If the account name is missing, the Initiator does not transmit the Account parameter when establishing a connection. Note that the user can use the account command to specify an account name for connection establishment. The format of the *.ftamrc* file is implementation defined.

-i turn off interactive prompting during multiple file transfers.

-g enable filename ‘‘globbing’’.

-d enable debugging.

-t enable packet tracing.

Commands

Note that BSFT commands and parameters are case sensitive. Where a command takes only constant parameter (e.g., the *type* command can take the parameter **ascii** or **binary**) it is sufficient to type enough of the parameter to identify it uniquely.

BSFT uses the parameter **ascii** in some commands. This has been used for compatibility with FTP. The files described as **ascii** are in fact text files.

account [*account-name*]
Define an account-name for the remote system (Responder). If the account-name parameter is omitted, BSFT prompts the user for a value. If the user replies to the prompt with no value, BSFT sets the default account value to an empty string, and does not transmit the Account parameter in subsequent connection establishment requests. This parameter may be required if the remote system complies with ISP 10607 but is not X/Open compliant.

append *local-file [remote-file]*
Append a file on the local system (Initiator) to a file on a remote system (Responder).

bell [**on** | **off**] Enable or disable a bell sound after each file transfer is complete. The use of this command without any parameter toggles the *bell* facility.

bye Terminate the file transfer session.

cd *remote-directory*
Change the working directory on the remote machine.

close Terminate the current association.

create *[password]*
Define a password for file creation. This parameter may be needed if the remote system complies with the base profiles but is non-X/Open compliant. If the password parameter is omitted from the create command, BSFT prompts the user for a password and does not echo the

value input by the user. If the user replies to the prompt with no password, then BSFT clears any password previously specified by the user and does not transmit a create password to the other system (Responder) in subsequent *put* or *mput* commands.

debug [**on** | **off**]

If the package supports a debug tracing facility then this may be enabled or disabled using this command. The support of debug information and, if supported, the type of information and the manner in which it is presented, are implementation dependent. The use of this command without any parameter toggles the *debug* facility.

delete *remote-file*

Delete a file on the remote system (Responder). If wildcard characters are included in the remote filename, BSFT behaves as follows:

a. if *glob* is *off*, prompt user for each filename with wildcards (*prompt* is *off* or *on*), or for any filename if *prompt* is *on*, or

b. if *glob* is *on*, prompt user only if *prompt* is *on* (for any filename).

dir *[remote-directory [local-file]]*

Display a listing of the specified directory on the remote machine; if the parameter *local-file* is present, then the directory listing is output to this file (if it already exists it is overwritten). If *remote-directory* is absent, the current directory is listed.

This command may not always be supported (it requires support of the optional NBS-9 document type) or if present may produce a listing with restricted parameters (only the filename is mandatory in the NBS-9 document type).

For each file in the filestore visible to the Initiator, a line containing six fields is shown. The lines are sorted using the name of the file as the key and in the collating sequence for the language being used by the Initiator. The fields are:

Field 1	Contents type indicator and Permitted Actions
Field 2	Identity of Creator
Field 3	Storage Account
Field 4	Filesize
Field 5	Date and Time of Modification
Field 6	Filename

For the Contents type subfield of Field 1, the following abbreviations are used:

Text	t
Binary	b
Directory	d

For the Permitted Actions subfield of Field 1, the following abbreviations are used:

Read	r

Insert	i
rePlace	p
eXtend	x
Erase	e
read Attribute	a
Change attribute	c
Delete file	d
Traversal	t
reVerse traversal	v
random Order	o

Note that the contents type values listed above are derived from the FTAM document types FTAM-1 (text), FTAM-3 (binary) and NBS-9 (directory).

An example of the output of the *dir* command is given below:

```
br-px-acdt--   creator   other     285   May 29   1987    .cshrc
br-px-acdt--   owen      admin     455   Feb 02   16:54   .ftamrc
tr-px-acdt--   root      root     4404   Nov 08   22:28   .kshrc
dr----acdt--   lak       bin      1024   Oct 26   17:41   bin
tr-pxeacdt--   ftam      osi      1081   Jan 17   18:28   record
```

encode (receive | transmit) *contents-type*

Define the encoding to be used with text files when receiving or transmitting a file. The parameter *contents-type* is defined by ISO in the FTAM standard and needs only to be specified when the Responder is not an X/Open system. It is one of the following:

ia5
graphicstring
visiblestring
generalstring

In addition, for *receive* only, the value *unknown* may be specified; this is the default value for *receive*. For *transmit* the default content type is a local matter.

get *remote-file* [*local-file*] [**-ascii** [*encoding*] | **-binary**]

Read a file from the remote system (Responder) and copy it to the current directory. The type of the remote file may be specified (otherwise the current default file type is assumed). If the file type is **ascii**, then the type of encoding to be used may be specified (see *encode* command); if not specified, the current default encoding is used. If *local-file* is absent, then *remote-file* is used as the local filename. It should be noted that *remote-file* may not be a valid UNIX filename because the responder may not be an X/Open system. If a local file with the name *local-name* already exists, or if *local-name* is absent and a local file with the name *remote-file* exists, then the local file is overwritten, or the *get* operation for the duplicate file fails completely depending on the override parameter.

If *local-file* is absent, and *remote-file* is not a valid name, then BSFT generates a new valid unique UNIX name from *remote-name*. The manner in which the new filename is produced is implementation defined; however, BSFT shows the corresponding remote and local filenames.

glob [**on** | **off**]

Enable or disable file wildcard expansion. To support globbing in parameters that specify files in the remote system the NBS-9 document type must be supported by both the Initiator (the user's system) and the Responder (the remote system). Some Responders may not support NBS-9, hence remote system filenames with wildcards may not be supported. In general wildcards are fully supported between X/Open compliant systems. (It should be noted that wildcards are always expanded by the Initiator.) The use of this command without any parameter toggles the *glob* facility.

help [*command*]

Display a short help message for the specified command. If no command is specified, *help* displays a list of all valid commands. The amount of help information and the manner in which it is presented is implementation dependent.

lcd [*directory*]

Change the working directory on the local system.

ls [*remote-directory* [*local-file*]]

Similar to *dir*, except that an abbreviated directory listing is output. It displays a simple list of the files in the filestore visible to the Initiator, one name per line. The list is sorted in the collating sequence for the language being used by the Initiator.

mdelete [*remote-files*]

Delete the files specified by *remote-files* on the remote machine. If wildcard characters are included in the remote filename, BSFT behaves as follows:

a. if *glob* is *off*, prompt user for each filename with wildcards (*prompt* is *off* or *on*), or for any filename if *prompt* is *on*, or

b. if *glob* is *on*, prompt user only if *prompt* is *on* (for any filename).

mdir *remote-directories local-file*

Similar to *dir*, except that multiple directories can be specified. Unlike *dir*, the parameters to this command are not optional.

mget *remote-files*

Similar to *get*, except that multiple filenames may be specified.

mls *remote-directories local-file*

Similar to *ls*, except that multiple directories can be specified. Unlike *ls*, the parameters to this command are not optional.

mput *local-files*

Similar to *put* , except that multiple filenames may be specified.

override (fail | overwrite)

Define the action to be taken when a file is sent to a remote system (Responder) and the specified file already exists on the remote system, or when a file is read from a remote system and the file already exists on the local system. The following values are allowed:

fail The write operation fails completely;

overwrite The file in the remote system is deleted and a new file is created.

open *host*

Specify the remote host. A connection is established immediately if auto-login is on (default), otherwise a connection is established when the *user* command is issued. If auto-login is on, the values for *user-name*, *password* and *account* is derived as described in the definition of the *-n* option, unless overridden by the *password* or *account* commands.

prompt [**on** | **off**]

Toggle confirmation for each operation. If *prompt* is *on*, then before a file is transferred or deleted by *mput*, *mget* or *mdelete*, the user is prompted for confirmation. The use of this command without any parameter toggles the *prompt* facility.

put *local-file* [*remote-file*] [**-ascii** [*encoding*] | **-binary**]

Transfer a file from the local system to a remote system (Responder). The user may specify the file as **ascii** (i.e., text) or **binary**. If the file type parameter is absent, then the current default file type is used. If the file type is **ascii**, the user may specify the encoding to be used; if not specified, the current encoding default is used. Since BSFT can interwork with both X/Open and non-X/Open systems, *remote-file* may specify an invalid UNIX filename.

pwd Display the current working directory on the remote machine.

rename *from-file to-file*

Rename a file on the remote system. This command may not always be available (it requires support of the File Management Service profile); this normally happens only if the remote system is not an X/Open compliant system.

separator char

Define a directory separator character for BSFT. This character will be automatically appended to the remote directory name specified by the *cd* command, before the remote filename is appended, to generate a full remote pathname. If this command is not supplied, the default value for *separator* is "/".

status Display the current status of BSFT. The status indicates the association currently established.

trace [**on** | **off** [*filename*]]

Enable or disable tracing of FTAM commands. The trace consists of the information that crosses the Presentation Layer interface. This facility is intended for system debugging purposes only. The information traced is implementation dependent, but should include the parameters defined in ISO 8571-3. Trace information may be directed to a file. The use of this command without any parameter toggles the *trace* facility.

type (**ascii** | **binary**)

Define a default file type for the transfer of files from the local system (i.e., the Initiator). If this command is not used, then BSFT determines the file type in an implementation dependent way. When receiving files, the file type is displayed when the read operation is completed provided verbose mode is enabled.

user *[user-name* [*password* [*account*]]]

Identify a user to the remote system and establish a connection to the remote system. The parameters of this command are mapped onto FTAM parameters as follows:

user-name	Initiator identifier
password	Filestore Password
account	Account

If the "account" parameter is omitted from the user command, BSFT does not prompt the user for an account, and instead uses the default account value (set by the account command). If the default account value is an empty string, BSFT does not transmit a value to the Responder. If the "password" parameter is omitted from the user command, BSFT prompts the user for a password (and does not echo the value input by the user). If the "user-name" parameter is omitted from the user command, BSFT prompts the user for the user-name and password parameters, as above.

verbose [**on** | **off**]

Enable or disable verbose mode. The use of this command without any parameter toggles the *verbose* facility.

! *command*

Run *command* as a shell command on the local machine.

Synonyms

For compatibility with current systems, the following synonyms should be supported:

quit	synonym for *bye*
recv	synonym for *get*
send	synonym for *put*
? [*command*]	synonym for *help*
binary	synonym for *type binary*
ascii	synonym for *type ascii*

Unsupported Functions

There are two functions, namely directory creation and deletion, that are not supported by BSFT and which users may see as complementary to the BSFT set of functions. These functions are not supported because the currently stable FTAM standards do not support directory operations. This deficiency is being addressed by the Filestore Management Addendum to ISO 8571, and BSFT will follow any recommendations that the profile groups may make on this work.

Appendix B

Requirements on the Lower Levels

The BSFT Specification requires that the OSI Connection Oriented Transport Service is used. The manner in which the transport service is provided to BSFT is outside the scope of the BSFT specification. Where, for portability reasons, a BSFT implementation is designed to make use of the XTI, this Appendix gives some advice. The XTI interface is defined in the **XTI Specification** (see **Referenced Documents**).

For interworking amongst BSFT systems and between BSFT and systems which support the base profiles, the only requirement on the lower levels is that all systems provide a compatible lower level stack. Work is underway in the profiling groups and ISO to define standard profiles for levels 1 to 4. Agreements on lower level protocols and profiles, and the choice of lower level options, are outside the scope of this document. The only requirement for BSFT conformance is that the Transport Service provides the services required by the Session Protocol as defined in ISP 10607-1 and summarised in **Section 9.4.**, **Session Layer**.

BSFT has a requirement to transfer session user data (data SSDUs) having a maximum size of 10 Kbytes. The support of this maximum limit is mandatory in ISP 10607-1. This may require:

- session service segmentation, or
- transport service fragmentation, and
  - — the ability to transfer fragmented TSDUs across the Transport Interface; this is implemented via the T_MORE Flag in XTI
  - — the support of large TSDUs across the Transport Service interface.

BSFT systems are not required to support Session segmentation. Hence, transport service fragmentation (which is mandatory in the ISO definition of the Transport Protocol) must be supported. The manner is which this is achieved, i.e., whether through the use of the T_MORE flag or large TSDUs, is an implementation choice. The T_MORE flag allows a TSDU to be passed across the XTI interface in several fragments. The support of this function is not mandatory in XTI and its use is not visible outside a system (provided the two ends support the same maximum TSDU size).

BSFT implementations are required to support the use of transport expedited to transfer ABORT, ABORT ACCEPT and PREPARE SPDUs (it is mandatory in ISP 10607-1). This can be supported through the XTI transport expedited data. For maximum compatibility, BSFT implementations may (as an option) also support the mode of operation where transport expedited is not used.

Appendix C

Glossary

ACSE (Association Control Service Element)

The ISO Application Layer entity that is responsible for establishing and terminating associations (i.e., co-operative relationships) between two applications.

ASN.1

Abstract Syntax Notation 1: a notation defined in the OSI standards that allows data to be described in a machine-independent manner.

Attributes (FTAM)

A piece of information stating a property of an FTAM file or an FTAM activity. The same value of a file attribute is observed at a particular time by any user of the file service, even if more than one user is active at that time.

CEN/CENELEC

Comité Européen Normalisation (CEN), and CEN Electrotechnique (CENELEC); these are the official standards bodies of the European Community (EC).

Constraint Set

A set of restrictions and refinements of a general file model which specifies a less general model tailored to the needs of a particular class of applications. BSFT supports only unstructured files, i.e., the 'Unstructured Constraint Set'.

CSMA/CD

Carrier Sense Multiple Access / Collision Detection: data link control protocol in which all stations share a common channel, and the collision detection mechanism averts clashes of simultaneous transmissions. The most common implementation is in Local Area Networks (Ethernet).

Document

A collection of information with known syntax and semantics and a known possible set of transfer syntaxes.

EWOS

Eurpoean Workshop on Open Systems.

File Access

The inspection, modification, replacement or erasure of part of a file.

File Management

The creation and deletion of files and the inspection or manipulation of file attributes.

File Transfer

A function that moves a part or the whole of a file's content between end systems.

FTAM
File Transfer, Access and Management: allows transfer of files between different operating system environments in a network, and also enables two cooperating systems in a network to allow a process on one system to access and manipulate files on the other.

globbing
Expansion of filename: if globbing is turned off, filenames are taken literally.

GOSIP
(Government OSI Profile) A government-defined network profile based upon a subset of the OSI protocol suite. It is designed to guarantee that all conforming implementations interconnect without problems.

Initiator
The FTAM file service element which proposes an FTAM association, and (in the case of BSFT) supports the user interface.

INTAP
Interoperability Technology Association for Information Processing, Japan, sponsored by the Japanese Government, Ministry of International Trade and Industry (MITI).

Internet Protocol (IP)
The protocol from the Internet Protocol Suite that provides the basis for communications over a large virtual network made up from a series of networks interconnected by routers.

ISP International Standardised Profile.

NIST National Institute for Standards and Technology" Division of the U.S. Government Department of Commerce that creates standards for use within U.S. Government areas. It was formerly known as the National Bureau of Standards.

OSI Open Systems Interconnection.

PCI Protocol Control Information.

PDU (Protocol Data Unit)
The data units exchanged by peer protocol entities. Examples are APDU (Application Layer), PPDU (Presentation Layer) and SPDU (Session Layer).

PICS Protocol Implementation Conformance Statement. A pro forma used by a user to state his protocol requirements or by an implementor to state his level of support for various protocol options.

Presentation
The OSI layer that provides for the representation of information that is communicated between (or referred to by) application processes.

Responder
The file service element which accepts the establishment of an FTAM association proposed by an Initiator. The Responder is responsible for mapping the VFS onto a machine's real filestore.

SDU (Service Data Unit)

The data units that are exchanged between a protocol entity and its users. Examples are ASDU (Application Layer), PSDU (Presentation Layer) and SSDU (Session Layer).

Service Primitive

The smallest defined interaction between a provider of a communications service and its user.

Session

The OSI layer that provides the merans to organise and synchronise the dialogue between application processes and manage their data.

TCP/IP

See *Transmission Control Protocol* and *Internet Protocol*.

Token Ring

A Local Area Network architecture in which a single token is passed from node to node, intercepted by a node waiting to transmit a message, held while the message is transmitted, and then passed onifollowing the end of the message.

Transmission Control Protocol (TCP)

The Internet standard transport level connection-oriented protocol. It provides a full duplex, reliable stream service which allows a process on one machine to send a stream of data to a process on another. Part of the Internet Protocol Suite.

Transport Service

The OSI layer that provides the transparent transfer of data between end systems.

VFS (Virtual Filestore)

An abstract model for describing files, filestores and possible actions on them.

Index